$2.00

kilobaud™

The Computer Hobbyist Magazine

ISSUE #1

January 1977

Articles

Is the Z80 the Wave of the Present? Pat Godding 20
Tiny BASIC . . . *a mini-language for your micro* Tom Pittman 34
How a Memory Works Reo Pratt 40
Software Exchange . . . *smoothing the rocky road* Art Childs 44
Practical Microcomputer Programming . . . *Part 1: Logical Instructions* John Molnar 50
Well, Your Micro's Built . . . *where do you grow from here* Lance Leventhal 54
Computer Control of the World!
. . . *turning ac powered devices on and off with your computer* Chris Bowick 62
Wire Wrapping . . . *Try it!* Dennis Brown 64
The Hobbyist's Operating System . . . *Part 1: Introduction and Master Plan* Dick Wilcox 68
Solving Some of the Software Interchange Problems Jef Raskin 76
Welcome to Assembly Language Programming Mike Aronson 78
Programming? It's Simple! Pete Stark 86
Structured Programming . . . *order out of chaos?* Bill Jones 92
Computers in Golf . . . *help for the handicapped* George Haller 96
Computer Widow Barbara Henderson 99
What's that Digital Group Really Doing? John Craig 100
How to Use the New PR-40 Printer Denis Bourdeau 104
Fire! . . . *let your micro call for help* John Craig 108
A Teletype Alternative Bob Grater 114
Nobody Knows the Troubles I've seen T. H. Lincoln, R. A. Walker, A. H. McDonough 118
Structured BASIC . . . *A negative view by Dr. Kemeny, the author of BASIC.* John Craig 122
Six Games on a Chip . . . *TV tournament time* Alan Dorman 130

Features

Publisher's Remarks 3
Editor's Remarks 4
News of the Industry 6
Letters 7
Books 11
Looking Ahead 13
The BASIC Forum 15
Around the Industry 16
Glossary 124

SwTPc NEWS

More Memory-Same Price
4K Now Standard In 6800

San Antonio–The SwTPC 6800 computer system, always a best buy is now an even greater bargain. Price reductions by the manufacturers of MOS memory circuits have made it possible to now offer the standard $395.00 6800 computer kit with 4K of memory instead of 2K as previously. Memory circuits are 21L02 types which make possible powering up to 24K of memory in the stock chassis with the standard power supply.

The Southwest Technical 6800 at $395.00 includes everything needed to work with your terminal. You get 4K of static MOS memory and a serial interface as part of the basic package. These are not extra cost options (?) as in many computer systems on the market.

8K MEMORY CARDS ANNOUNCED –

For those 6800 systems needing the maximum possible amount of memory, Southwest Technical Products announces 8K memory cards. These memory expansion cards have 8K Bytes of low power MOS memory per board. These kits feature the new 4K static RAMS that are now becoming available. These new RAMS make it possible to put 8K of memory on a board without crowding the parts, or using small hard to solder connecting lines. These new memory boards feature DIP switch address selection and a write protect switch on each board.

The low power consumption of this new memory board makes it possible to use up to 48K of memory in the standard 6800 chassis with the stock power supply. Priced at $250.00 these memory cards cost no more than less dense memories from other sources.

PRICES CUT ON 4K MEMORIES

Southwest Technical Products has reduced the price of its standard 4K memory card by 20%. These cards use low power 21L02 static memories. The new price for the MP-M memory kit is $100.00 for a full 4K kit.

This kit contains 4K of memory with full buffering and dual on-board voltage regulators. Six of these memory cards may be used in a standard 6800 chassis to provide 24K of memory for the system. Memory now becomes even more of a bargain–24K for only $600.00.

Who Needs It?

We continue to get reports from customers who are amazed at the ease of assembly of the 6800 computer. One reports that he purchased test equipment before ordering a computer at the advice of friends who owned brand "X" machines. His total use of the test equipment was zero (0) when he installed each board in the 6800 and they all proceeded to work perfectly the first time. He later found in comparing notes with other 6800 owners that his was not a unique experience.

People who have built most of the various types of computers on the market generally agree that our instructions are the best and most complete. So don't worry about purchasing the least expensive computer system, there are still good honest values being offered in the world of personal computing.

SUPER SOFTWARE

"Lack of Software" can no longer be used as an excuse by those who have the poor taste to purchase computers using older, less elegant processors than the MC-6800. Southwest Technical Products has not only editor-assembler and game programs available for the 6800, but also both 4K and 8K BASIC.

The ability to run ANSII standard BASIC programs on the 6800 make the enormous number of BASIC programs out there all usable on the SwTPC 6800. That's right, you can run anyones BASIC programs on the 6800 provided they are written in standard BASIC (as most are). 4K Basic at $4.95 and 8K BASIC at $9.95 are inexpensive enough for anyone to own. They do not cost hundreds of dollars as in some systems, or only become available when combined with purchase of huge amounts of memory as in others.

Loading even a relatively long program such as 8K BASIC into your SwTPC 6800 is not a long procedure when the AC-30 cassette interface is used. This super reliable and inexpensive ($79.95 complete with cabinet and power supply) cassette interface uses the "Kansas City" standard format and will load 8K BASIC in approximately five minutes.

SOUTHWEST TECHNICAL PRODUCTS CORPORATION
219 W. Rhapsody
San Antonio, Texas 78216

DIGITAL DATA RECORDERS



USING 3M DATA CARTRIDGES



Model 3M3 – $199.95

Model 3MI – $169.95

BRAND NEW DESIGN!

Featuring the radically new "UNIBOARD" method of construction for data cartridge drives. The major computer manufacturers are changing from cassettes to cartridges at a rapid pace because of freedom from binding and greater data reliability. Now, these professional type units are priced within the range of all data users. Being made primarily as OEM data storage units for the world's major manufacturers, these units, together with controller board and software ROM, are being made available to the individual user as well.

*Appearance and specifications may be changed slightly following acceptance tests now being conducted by OEM users.

MODEL 3M3 Uses the 3M Data Cartridge, model DC300. This cartridge contains 300 feet of .250 tape in a sealed container. Records and plays at 9600 baud NRZ, 4800 baud P.E. Nominal speed 8" per second. Max. recommended flux density 1200 fcpi. Using four tracks, you can store nearly 2 megabytes of data on a cartridge. Cartridge measures 4" by 6". Turns counter indicates tape position. Inter-record gap light gives more accurate position 2SIO(R) is NOT required for use but is highly recommended for 8080 and Z80 systems.

COMMON SPECIFICATIONS FULL SOFTWARE CONTROL of record, play, fast forward and rewind. LED indicates inter-record gaps. EOT and BOT are sensed and automatically shut down recorder. Can also be manually operated using the switches on top which parallel the software control signals when not under software control. Signal feedback makes it possible to software search for inter-record gaps at high speed. 117V – 60 Hz – 5 watts.

TWO I/O PORT CONTROLLER WITH ROM Controls your terminal and one or two cassettes or cartridge units. On board ROM (for 8080 and Z80) has terminal and cassette software for turn on and go operation. NO MORE BOOTSTRAPPING. Plug in compatible with Altair and IMSAI. Loads and dumps memory in Hex from the keyboard, formats tape files, punches tape, functions as a word processor and searches for files and four letter strings within files. Keyboard controls the cartridge units above on rewind and fast forward. Special keyboard codes enable you to dump and read Phase Encoded tapes as well as NRZ tapes. (Including K.C. Std.) Call routines give access to these from your software.

MODEL 2SIO(R) – (Assembled & Tested) With 1 ROM for NRZ Cassettes $169.95 (Half of above Program)
With 2 ROMs for Data Cartridges and P.E. cassettes. $189.95 (Full Program)

Kits available for $30 off above prices.

MODEL 3MI – Uses the 3M Data Cartridge type DC100A. This cartridge contains 150 feet of .150 tape and is the same cartridge used by H.P. and others. Runs at 4800 baud NRZ, 2400 baud P.E. Tape speed adjustable but nominally set at 5"/second. Maximum recommended flux density 1200 fcpi. Cartridge measures 2-1/8" by 3-1/4". This model is ultra compact, yet extremely capable. It is intended for word processing, mailing list use and other applications requiring the compact storage of data. Data location is by inter-record gaps and automatic file search. See Common Specs and 2SIO(R) below. 2SIO(R) is NOT required for use, but is highly recommended for 8080 and Z80 users.

For 8080 and Z80 users: Comes complete with software program listings for the programs on the 2SIO(R) ROM below. 6800 software is being written but not yet completed. These programs give FULL SOFTWARE CONTROL.

CARTRIDGE AVAILABILITY Cartridges are made by 3M, ITC, Wabash and others. They are available at all computer supply houses and most major computer service centers. We can also supply them at normal current list prices.

NEW AUDIO CASSETTE INTERFACE* Phase Encoding interface for use with audio cassettes or NRZ recorders. Runs 2400 baud phase encoded on good quality audio cassette recorders. May also be used with 2SIO(R) above to use the 2SIO(R) cassette programs with your audio cassette player. Can also accommodate "Tarbell" tapes and K.C. Std. tapes. $50.00, Wired & Tested.
$35.00, Kit Form.

*NOTE: You do not require an interface with the 3M1 and 3M3 unless you Phase Encode. But, you do need an interface to use the 2SIO(R) with your own audio cassette.

"COMPUTER AID" and "UNIBOARD" are trademarks of the NATIONAL MULTIPLEX CORPORATION. The 3M Data Cartridges are covered by 3M Patents and Marks. "UNIBOARD" Patents Pending.

OVERSEAS: EXPORT VERSION – 220 V – 50 Hz. Write Factory or – Datameg, 8011 Putzbrunn, Munchen, Germany; Nippon Automation 5-16-7 Shiba, Minato-Ku, Tokyo; EBASA, Enrique Barges, 17 Barcelona, Spain; Hobby Data, SpireaVagen 5, Malmo, Sweden; G. Ashbee, 172 Ifield Road, London SW 10-9AG.

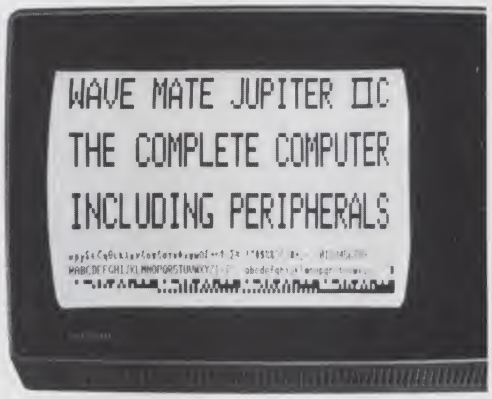
For U.P.S. delivery, add $2.00 each item. Overseas and air shipments charges collect. N.J. Residents add 5% Sales Tax. WRITE or CALL for further information. Phone Orders on Master Charge and BankAmericard accepted.

BANKAMERICARD master charge

NATIONAL MULTIPLEX CORPORATION
3474 Rand Avenue, South Plainfield NJ 07080, Box 288. Phone (201) 561-3600 TWX 710-997-9530.

Now we're on TV!

Wave Mate introduces Jupiter IIC, a complete computer system incorporating a monitor quality TV interface. This system provides everything you need to create and run application programs. Jupiter IIC includes a CPU with 8K dynamic RAM and 3K ROM memory, video terminal interface and keyboard, and dual audio cassette tape interface. The TV interface features upper and lower case and Greek character sets, and dot graphics. The dual audio cassette interface provides start/stop operation and operates at 300, 600, or 1200 baud. And of course we still provide these high-quality features: burn-in tested IC's, socketed IC's, complete documentation, and more.



ATTENTION: ORIGINAL EQUIPMENT MANUFACTURERS

Jupiter IIC provides OEMs with the tools to get systems into the field faster and at lower cost. (1) Use Jupiter IIC as your development system. Perfect for development of software and special hardware. (2) Use Jupiter IIC for prototype systems. Only Wave Mate provides the tools – wire wrap modules, universal modules, complete documentation – to easily tailor system logic and add customized interfaces within the basic Jupiter IIC package.

SOFTWARE

All Jupiter IIC systems feature a sophisticated monitor/debugger package including a versatile interrupt system and I/O monitor call instructions. A programmable macro editor and expanded assembler are also provided. Proposed ANSI standard BASIC is included with Jupiter IIC.

THE JUPITER IIC KIT: $2850

The kit includes the CPU, software debugger and monitor module, 8K dynamic memory, module cage, power supply, front panel, video interface, cassette interface, and all the documentation required to assemble, run, and understand the system as well as modification instructions for a black and white TV set.

THE JUPITER IIC ASSEMBLED SYSTEM: $3800

All components of the Jupiter IIC kit plus two audio cassette units and a 12-inch black and white TV set. The complete system is shipped with all components assembled and tested.

SPECIFICATIONS

CPU
MC 6800; eight-level interrupt, prioritized and maskable by level; single-cycle and block DMA

DUAL AUDIO CASSETTE
Complete paper tape replacement; start/stop motor control; 300, 600, or 1200 baud (crystal controlled); error correction

VIDEO TERMINAL INTERFACE
64 x 32 lines
Upper and lower case, plus Greek alphabet; 7 x 12 format, 128 dot (hor.) x 96 dot (vert.) graphics

MEMORY
8K dynamic RAM; 3K ROM; 1K dual-port static RAM

KEYBOARD
Generates full 128-character ASCII set



☐ Send details on Jupiter II systems
☐ Have salesman call

Name _____
Title _____
Company _____
Address _____
City _____ State _____ Zip _____
Phone _____

Wave Mate

WAVE MATE 1015 West 190th Street, Gardena, California 90248
Dept. 203

Telephone (213) 329-8941

1 Print "Publisher's Remarks" 10 No End Run

Wayne Green

What Will *Kilobaud* Be Like?

Tell you what — first I'll tell you what *I* want in a computer magazine and then you tell me what *you* want — for the fact is that *Kilobaud* will be mostly what you want, but I won't know what that is until you respond.

I look upon computers as fun — a hobby. But I'm by no means oblivious to the coming small computer market. I realize that if I am going to be in a position to take advantage of the market to come I need to know all I can about computers — how to design them — how to program them — how to use them — how to service them. I have to know what hardware is available and what it will do — what peripherals will work with what systems — what sources there are for programs — what test equipment I need — things like that.

The primary thing I expect from a computer magazine is that I be able to understand it. If I don't understand articles, that is not my fault, but the fault of the editor. I don't want to be patronized and I don't want to be talked down to; I just want to learn as much as I can and have fun while I'm at it.

This brings me to my concept of *Kilobaud*. I don't know how much of it will stick, we'll see. My idea is to publish a computer magazine which will make it possible for newcomers to computing to get up to speed. It is all too easy to get so involved with advanced ideas that you forget the beginner — we've done this a bit with *73* in recent years and I'm working hard to get back to where some of *73* is for Novices.

I view *Kilobaud* much as I do *73* — as a medium for hobbyists to contact hobbyists — sort of a large scale newsletter. We'll try to keep the editorial ego trips down and go light on the avuncular advice (except in the editorials and answers to letters).

You, the reader and hobbyist, are a prime contributor to *Kilobaud*. Please make sure you keep an accurate log of your work with your computer system, making note of all problems you encounter and the solutions to same. There may be parts missing from a kit — lousy or perhaps misread instructions — faulty parts — whatever. If you'll write up your adventures for the letters section of *Kilobaud*, a lot of people will be able to benefit from your experiences. I'll try to keep such phrases as "reinventing the wheel" and "bells and whistles" out of *Kilobaud* . . . with your cooperation.

I think we can save each other an awful lot of time and misery if we pass along the word on our problems and the solutions. It could result in some pressure on recalcitrant manufacturers to get their acts together too.

Having been a manufacturer (loudspeaker enclosures — I had seven factories going strong making them), I'm familiar with their problems . . . and there are a lot of problems.

In order for us to understand more about computers, I'm trying to get the manufacturers to write articles telling us about their systems — why they were designed the way they were — what the benefits are to us, the users, and what the trade-offs were. Perhaps this series will take some of the mystique out of computer design and free us from thinking of the CPU as a black box. If we're going to learn to service these gadgets we really have to come to grips with them.

Yes, I know manufacturers tend to think very positively about their systems, but they have to create credibility too — and that takes a good deal of candor. I don't think many of us are going to be hoodwinked — particularly steely-eyed Editor Craig.

I doubt if any of us are unaware of the sad state of software for our toys. As a matter of fact, it is this lack which keeps them toys instead of hot little money-makers. I don't know about you, but one of my local druggists has already been asking about getting some time on my system. He'd like a way to quickly look up customers vs. prescription numbers and would be willing to pay for renting a terminal and phone line just to have that dope at hand. Hmmm.

Okay, all of us need software — it may be possible to coax a good deal of it from *Kilobaud* readers who would like to get credit for writing some neat programs and would like to help others — plus the not inconsiderable impetus of the healthy payments for articles in *Kilobaud* which runs about double that in any other computer magazine (as far as I know). A couple of good programs could go a long way toward financing a new microprocessor.

My concept is to publish the shorter programs and routines in *Kilobaud* so all of us will have a continuing library to use when we want to write a new program. We'll be working hard to standardize programming techniques toward the end of making program modules of value. Longer programs will have to be handled differently and my idea is to publish and sell them much as we do books, complete with a royalty to the writer.

One of the major problems with selling software is theft. I've found that if you put something out at a reasonable price (a *reasonable* price is one that seems reasonable to the buyer, not the seller) you have little problem with theft. I went to incredible troubles a couple of years ago to make a tape master of a series of cassettes teaching the Morse code and basic radio theory. Since then we've been selling these cassettes at a brisk rate and run into very little in the way of copying. The reason is that we sell the tapes for $4 for a one hour tape, and it just isn't worthwhile to make copies.

I am sure that if we are able to sell programs at low prices and depend upon volume to bring in the profits, we'll have little problems with theft. I envision cassettes of programs mounted on cards and complete with good documentation, hanging by the hundreds from hooks in computer stores, selling from $2.50 to maybe $50 for extensive business application programs. With hundreds (or perhaps thousands) of computer stores selling the programs the end result should be a tidy royalty.

Let's say we have a nice inventory program which would be applicable to stores with a lot of products such as shoe stores or book stores. If this were to sell for $19.95 in the computer store, it would then have to sell to the computer store for about $12 to permit the necessary profit margin which would pay for their rent, advertising, service, sales help, cataloging, and so forth. The royalty would be 15%, which is $1.80 per program sold. A program like this would be of interest to virtually every businessman and to most hobbyists (who could then rent time on their systems locally). How much could we sell to a market of 100,000 hobbyists and 50,000 businessmen? If we only sold to 10% this would amount to $27,000 in royalties. If we have a market anything like I think we will, with 1,000,000 businesses and who knows how many hobbyists and home users, what could be sold? Whoooee!

Before you get all enthused about putting out your own programs on cassette, let me acquaint you with the facts of life of manufacturing. On our $19.95 net program the manufacturer

continued on page 14

kilobaud

EDITOR
John Craig

PUBLISHER
Wayne Green

MANAGER
Biff Mahoney

PRODUCTION MANAGER
Ruth Brown

EDITORIAL ASSISTANT
Kurt Schmidt

ASSOCIATE EDITORS
John C. Burnett
Warren Elly
Rich Force
Stan Miastkowski
Susan Philbrick

PRODUCTION DEPARTMENT

Manager:
Lynn Panciera-Fraser

Staff:
Craig Brown
Gayle Cabana
Robert Drew
Michael Murphy
Bob Sawyer
Noel R. Self
Robin M. Sloan
Jody Wright

PRINTING
John Bianchi
William Cering
Brent Lawler
Michael Potter
Gary Steinbach

PHOTOGRAPHY
Bill Heydolph

TYPESETTING
Barbara J. Latti
Sandy White

MARKETING
Sherry Smythe
Karen McDonough

ADVERTISING
Bill Edwards
Leslie Bailey
Nancy Cluff
Janet Ames
Lisa Healey

COMPTROLLER
Knud M. Keller

ASSISTANT COMPTROLLER
Marge Nielsen

ADMINISTRATIVE ASSISTANT
Regina Phillips

COMPUTER ENGINEERING
C. Robert Leach
David E. Wilensky

CIRCULATION
Dorothy Gibson
Barbara Block
Judy Brumaghim
Susan Brumaghim
Nancy Chandler
Cher Dean
Janette Geoghean
Fran Dillon
Florence Goldman
Pat Hogan
Lois Ireland
Peggy Sysyn
Theresa Toussaint
Judy Waterman

INVENTORY CONTROL
Marshall Raymond
Larry Ames
Gary Slamin

PLANT MAINTENANCE
Bill Barry
Lorraine Pickering

DRAFTING
Bill Morello
Carol Ann Beverlie
Lynn Malo

Kilobaud is published monthly by 1001001, Inc., Peterborough NH 03458. Subscription rates in the U.S. and Canada are $15 for one year and $35 for three years. Outside the U.S. and Canada, add $2 per year. Application to mail at second class postage rate pending at Peterborough NH 03458 and additional mailing offices. Phone: 603-924-3873. Entire contents copyright 1976 by 1001001, Inc. INCLUDE OLD ADDRESS AND ZIP CODE WITH ADDRESS CHANGE NOTIFICATION.

20 Print "Editor's Remarks"
30 End
Run

John Craig

Kilobaud. The name brought forth smiles, moans of despair, laughter (even side-splitting on occasion), grins, anger, and cheers. In a few (very few) instances it was received with apathy. Which was okay, because if there was one thing we didn't want, it was an apathetic name! But, what the heck. A name doth not a magazine make. You're not going to be reading these pages month after month because of a name, right? No, the incentive for that will most surely be provided by the content.

As *the* "high capacity data channel" for the personal computer community, *Kilobaud* will be doing it's utmost to bring you the latest, most practical, interesting, and *provocative* material to be found anywhere. And, just so there's no mistake regarding our purpose, let me restate it: *Kilobaud* intends to promote the fun and practical application of personal computer systems.

One of the ways we're going to accomplish this is by providing a means for the beginner to get started and keep going with this hobby. But let's stop right there for a moment and define "beginner." I'm referring to the novice, the person who has never been exposed to a computer, *and at the same time* to the experienced computer type. There is the hardware type who is a "beginner" in the software area and could use (and wants) some good programming techniques, tricks, and tools to help him along. I'm also referring to the experienced software type who is a "beginner" when it comes to flip-flops, pull-ups, power supplies, Tri-state* logic, cross-coupled NANDs, and all the other glitch generators we call hardware. In most cases that programmer is going to be just as interested in how to debounce a switch as the hardware type is in doing a shell sort. Which brings up two very important points. Number one, we're going to do everything possible to ensure that we're all talking the same language. In other words, the "buzzwords" will either be eliminated or defined in a glossary contained in each issue. (A couple of prime candidates might be "debounce" and "shell sort.") Number two, we're going to accomplish all this by writing for each other. The programmer will, of course, have to sit down and write an article for the hardware type on using a shell sort. He, in turn, will have to explain a debounce circuit to the programmer.

Now . . . to elaborate on that last point. You're not going to see articles in *Kilobaud* where that programmer author simply says, "Here is a shell sort." Nor are you going to see one where that hardware author says, "Here is a debounce circuit." No, I'm afraid that approach would be a little dry. The hardware type needs to explain to us software-oriented types what happens during data entry with a "noisy" switch (i.e., *why* is the debouncing important), several methods or circuits to accomplish the objective (with a discussion of why one circuit is better or worse than another), the nuts-and-bolts implementation of the circuit, and any other practical considerations. Likewise, the programmer is going to have to tell us hardware types *why* we would want to use a shell sort. What are the advantages? How does it compare to other sort routines? Is one more suitable for certain applications than the other? And of course the nuts-and-bolts again . . . in this case in the form of flowcharts and the actual program.

Another way *Kilobaud* will strive to get people turned-on to personal computers is by providing articles dealing with applications. We're going to have articles dealing with the many possible home applications (entertainment, education, accounting, environmental control, etc.); but, once again, the material will be presented in a down-to-earth manner which we can all understand and get the most out of. (Quite some time ago I was told by the editor of another magazine in this field that if I wanted his respect I would make sure the articles I selected were as technical as possible. I won't bother you with the details of my reply; but I did want to point out that his magazine has an approach which is different from ours . . . and if you're looking for the "heavy" stuff, look there.)

Speaking of writing . . . it's interesting that, in view of the fact we've made such a big deal out of *Kilobaud* being a non-Ph.D. magazine, we seem to have wound up with a lot of Ph.D.s writing for the magazine. Needless to say, they're not writing for Ph.D.s . . . they all have that ability to get down to the brass tacks and explain things in everyday, easy-to-understand terms. That's really the only criteria for *you* to write for *Kilobaud* too. We're not particularly looking for professional writers. *Kilobaud* is going to be a loose, informal hobby magazine just like *73 Magazine* is for amateur radio enthusiasts and electronic experimenters. If you're toying with any thoughts about writing for *Kilobaud*, I feel it's only fair to warn you about the *money*. We pay out lotsa bucks for good material . . . so be prepared!

Where's it all going?

If you're thinking this personal computer "movement" is going to remain in it's current state (i.e., limited to several thousand enthusiasts) *you're wrong*. We're going to see this thing bust wide open in the near future. For example, a couple of multimillion-dollar companies are going to introduce home computer systems in 1977. One of these companies will use (from what I've been told) nationwide TV ads to sell their system.

Stop and picture the following situation for a moment: You are Mr. and Mrs. Average American who still consider a computer as nothing more than a mistake-prone monster. It's Tuesday evening at 8 pm and you're settling back in the ol' easy chair to watch some TV. A commercial comes on. A commercial for a home computer system? What next! But wait, this looks interesting. They're showing you how easy this thing is to use (i.e., bring it home and turn it on) and some of the interesting things it can be used for. There are some really neat entertainment programs available: chess, checkers, video games (look, there's "TANK"!), Qubic, as well as some practical programs for around the house, such as an accounting and tax preparation program and index programs for magazine articles, recipes, etc. Also demonstrated in this 60 second ad are some educational programs which include math exercises.

Now, since you're one of the millions of gadget-conscious Americans, and assuming the ad convinced you "that you just have to have one of these," there's a good chance you're gonna go out and buy one! There are three things which will convince the American public they should have a home computer: 1) The ease with which it can be used (i.e., not having to learn to become a programmer, being able to simply load a program in from an ordinary cassette recorder, etc.); 2) The uniqueness and practicality of having one (which could be a problem; but I don't think so if some good application programs are offered); 3) And last, but not least, the cost. And that's the clincher. It's so low that I've been sworn to secrecy. Wait til you see it!

The above scenario *will* take place. And, if it's not with the company I'm thinking of, it's just a matter of time before another one comes along and does it. The potential side effects are mind-blowing! The number of boards built for the Altair bus is staggering; but you ain't seen nuthin' yet! Just wait until companies start jumping on the bandwagon with this one. (We may even have some further complications thrown into the works by a "battle" shaping up over which bus will become the standard.)

All of that brings up another thought. The biggees have begun to take notice of what's going on. Digital Equipment, for example, recently came out with their Direct Sales catalog for the single-quantity purchases of the hobbyist. Plus, their LSI-11 has had some success within the hobby community . . . and they'd like to see it increase. Another interesting point regarding DEC and the hobby community is that I recently heard they are going to develop an Altair bus interface so their customers can take advantage of low-cost memories and other goodies available. Then we have National Semiconductor. Without a doubt, they're making a thrust into the hobby market. And, I think we could put Motorola into the pot also. In the months to come I'm sure we'll see even more of them waking up.

The Computer Faire

Ahh . . . there's nothing like a faire in the springtime. And we're going to have a beaut this spring! *The first west coast computer faire* will be the first major convention held on the west coast for the hobbyist community, and even more significantly, it will be somewhat of a "homecoming" since it's going to take place in the San Francisco area . . . which is where most of it all began (in the "Silicon Valley"). Jim Warren, editor of Dr. Dobb's Journal, and Bob Reiling, editor of the Homebrew Computer Club Newsletter, make up the brawn and brains behind this venture; and with their collective talent behind it we should have quite a show.

Following is a list of major topics for seminars and informal talks which are being planned:

- Hardware, Software & Systems for Home Word Processing
- Speech Synthesis Using Home Computers
- Computers & Systems for Very Small Businesses
- Microprogrammable Microprocessors for Hobbyists
- Digital Cassette Tape Standards

*Registered trademark, National Semiconductor Corp.



- Program & Data Input via Optical Scanning of Bar Coded Information
- Personal Computers for Education, which will have associated with it, a University of California short-course
- Computer Graphics on Home Computers
- Computer-driven & Computer-assisted Music Systems
- Personal Computers for the Physically Handicapped
- Computers & Amateur Radio
- Peripherals Interface & Bus Standards
- Software Design, Modularization & Portability
- Floppy Disc Systems for Home Computers
- Computer Games — Alphanumeric & Graphic
- Discussion Sessions for Computer Club Officers, Convention Organizers, Club Newsletter Editors, etc.

Other *Conference Sections* will be added as topics are proposed and speakers are found.

Jim and Bob are looking for speakers for the areas listed above; so if you're interested, be sure and drop them a line for further information. (I'll be giving a talk on Low-cost Small Business Systems.) Contact: Jim Warren, Faire Chairperson, Star Route Box 111, Woodside CA 94062, (415) 851-7075; 851-7664; 323-3111 or Bob Reiling, Operations Coordinator, 193 Thompson Square, Mountain View CA 94043, (415) 967-6754.

The First West Coast Computer Faire will be held in San Francisco Civic Auditorium on April 15, 16 and 17, 1977.

Where's all the software?

Why are people still sitting around developing language processors and editors for the 8080 and 6800? I'll tell you why. Even though these packages have already been developed, a large number of people got into this thing as a hobby so they could develop just that kind of software (for fun). I can relate to that . . . but darn, I sure wish they'd pour their efforts into the development of some good applications software! (I mean, we could sit around doing the same old thing for the next five years!) None of us ever stop hearing that question, "Yeah, but what do you use it for?" Aside from answering that it is simply our toy and we enjoy it as a hobby, it would be nice to have the little monster doing some practical things. Not only to impress the neighbors, and ourselves, but to help justify the thing to the lady of the house.

And what's all this jazz about the Intersil 6100 being able to run PDP/8 software? I haven't heard of anyone really going to town on this one. If I have a 6100 machine with a paper tape reader interfaced, can I take a PDP/8 program and read it right in and run it? What are the constraints, if any? Will *all* PDP/8 software run on the 6100? What about the restrictions regarding DMA, and how can they be overcome? What are the mechanics involved in getting hold of the vast amounts of PDP/8 software which has been developed over the years? And what about the cost? Sounds like material for what could be one of the most significant articles anybody has seen in a long time. (*Last minute flash* . . . just got a phone call from Steve Diamond of Intersil, and he's going to answer those questions in an upcoming article.)

If you've written any good programs that you'd like to share with the rest of the world, then how's about submitting them to the *Kilobaud* Software Library where they will be exposed to nationwide distribution? *Kilobaud* will pay a healthy 15% royalty to the author/programmer also. In the second issue of *Kilobaud* we'll have the criteria for submitting programs to the library. It should be pointed out that you don't have to be a top Grade A professional or semiprofessional programmer to submit programs to the Library or for an article in the magazine. Just so it runs. You're going to be seeing programs in the pages of *Kilobaud* written by beginners. (Those of you who aren't, at least remember what it was like, right?) They're not going to be perfect programs, and in many cases I'm hoping the professionals among our ranks will drop a letter to the editor (or whatever) and point out where the author went wrong . . . and just as important, how it should have been done. In other words, we're going to do everything we can to encourage people to sit down and develop programs *and* write about them. I certainly don't want to turn anyone off (newcomer or professional) by rewriting their programs to reflect my style or interpretation.

Looking back

Sometimes it's kind of fun to take a look back and see where we've been, and in some cases it gives us better perspective on where we're going. Since the computer hobbyist movement is such a new thing, we're not going to be able to go back very far. You know, aside from a handful of home brewers, this whole thing is only 2½ years old . . . having been started by Jon Titus' article in the June '74 issue of *Radio Electronics* on building the Mark-8 minicomputer. The Mark-8 was an 8008-based microcomputer which had some problems (which most people managed to overcome); but looking at the big picture, and what it started, I'd say it was quite a machine! Let's go back and see what things were like *two years ago* in January of 1975:

— The recently formed *digital group* in Denver CO was offering (for $7.50!) complete schematics and software developed by Dr. Robert Suding for interfacing a cassette recorder to the Mark-8, along with a cassette containing a bunch of useful programs: cassette load/dump, TVT & Keyboard interface, memory tests, and more.

— An 8008 microprocessor chip was selling from $50 to $60 (not too many people were even considering the 8080 at the time because of its $175+ price tag). 1702 PROMs were selling at $40, and 1101 RAMs (that's 256 x 1 bits, folks!) were selling anywhere from $3 to $5. 2102 RAMs (w/8008 interface) was Creed TTYs selling for $145. (As a matter of fact, the CREED is still alive and well. See Art Child's article on paper tape devices in next month's issue.)

— The hobby world was in deep wonderment and consternation over the recently announced Altair 8800. Apparently MITS wasn't expecting the response their new baby brought forth, and as a result they were caught with their guard (and delivery schedules) down. In retrospect it looks as though they came out of i pretty well. It's for sure their . .ct has been felt within the hobby community . . . and will very likely continue to be felt in the future.

— CACHE (The Chicago Area Computer Hobbyist's Exchange) had its beginning about this time. The first Chicago club was started by Robert Swartz, Bob Cook, Mark Condic, and Don Martin of Martin Research. Don's 8008-based MOD-8 computer was probably the 2nd in popularity with the hobbyist two years ago. He was really after the industrial market, though, and published a book entitled *Microcomputer Design* which sold for $100 (or $120 with an 8008 chip). (It was printed on red paper to discourage copying!)

— The fifth issue of the Micro-8 Newsletter was about ready to be mailed out by Hal Singer and John Craig (that name sounds familiar). Incidentally, Hal still has back issues of the Newsletter available at a nominal fee. Why don't you drop him a line at the Cabrillo Computer Center, 4350 Constellation Road, Lompoc CA 93436 (otherwise, he's going to wind up using those things to start his fireplace in the years to come).

The Kilobaud Glossary

As a regular feature each month, we're going to publish a glossary which will define any and all buzzwords we spot in the articles for that issue. There is, of course, the rare possibility that we could make a mistake and let one get by us. If that happens, be sure and drop us a line and tell us about it. As a matter of fact, if you should happen to run across new phrases or words any you discover the meaning for, then drop me a line about it and we'll share it with the rest of the world.

I should point out to potential authors that the glossary doesn't mean a green light to freely use buzzwords throughout your articles. Not at all! Well, let me clarify that . . . you can use them all you want . . . just explain them in the text. The glossary is primarily for catching the "loose" ones. ■

KILOBAUD SWEEPSTAKES DRAWING



Pretty ten year old Diane Mathieson of Trenton, New Jersey drew the winning card from our 73/KB sweepstake's box at PC 76 while Jim Muehlen looks on with anticipation. The lucky winner, Byron Young, Jr. of Pasadena, Texas, has a Windjammer Cruise of the Caribbean for two in his future.

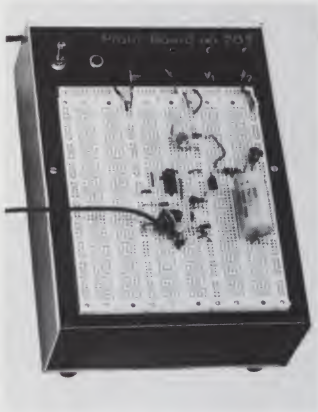
NEWS OF THE INDUSTRY

NEW PRODUCT ANNOUNCEMENTS & REVIEWS

Kilobaud will be doing its best to keep you abreast of the latest new products as they apply to the hobbyist market. You won't be seeing announcements here for two, three and four thousand dollar processors and peripherals because our feeling is that equipment in that price range was developed for industrial/commercial users, rather than the personal small system user. We would like to review each new product that comes on the market to give you a good objective evaluation of it; but that simply can't be done. (This is why we're so anxious to see articles or letters written by hobbyist consumers describing their experiences with something they've built and/or used.) We will also make every attempt to establish the integrity and reliability of companies sending us new product announcements. — John.

Continental Specialities Corporation Proto-Board

It would be pretty tough to come up with an easier way to design circuits than CSC's new Proto-Board. Models range from the PB-6, with 360 solderless tie-points (in kit form for $15.95) to the PB-203A with over 2250 tie-points and built in ½ amp +15 and -15 volt adjustable regulated power supplies (selling price $120.00).



All models are set up for interconnection, so you can multiply for greater capacity. Terminals are 5-point, with the manufacturer claiming a thousand applications. The Proto-Boards are not only the newest thing in breadboards, but an experimenter's delight.

For further details, contact Continental Specialties Corporation, 44 Kendall Street, Box 1942, New Haven CT 06509.

Continental Specialites Corporation Proto-Clips

Continental has added a 24-pin Proto-Clip to its line, after strong success with their 14- and 16-pin models. There simply isn't an easier way to bring leads up from crowded circuit boards or to wire unused circuits into existing boards.

The clips are one piece high-impact plastic, eliminating springs or pivots. Contacts are noncorroding nickel-silver, and there are Proto-Clips available with prewired cables, in the 14-, 16-, and 24-pin configurations.

One of the best features of the clips is their cost. Only $8.50 for 24-pin models, less than $5.00 for 14- and 16-pin clips. A must for computer hobbyists' troubleshooting kits.

For additional data, contact Continental Specialties Corporation, 44 Kendall Street, Box 1942, New Haven CT 06509.

Advanced Data Sciences Univue Enclosure

Looking for a place to put your keyboard, keypad, video display generator, or microprocessor? Advanced Data Sciences has just what the doctor ordered. It's a two piece, aluminum and steel, low profile enclosure, with over 200 cubic inches of space. Included in the package is a flush mounting aluminum front panel, which can be easily scribed and drilled for mounting purposes. Since it comes off the enclosure with six self-taping screws, the panel can be spray painted to match your equipment, making for a clean, professional package. Surface area of the panel is 23" X 8".

Aside from microprocessor applications, the Univue is also applicable to keyboard Morse, RTTY, and a host of other uses. Including rubber feet and mounting hardware, the Univue sells for $32.95 plus postage and handling from Advanced Data Sciences, PO Drawer 1147, Marion OH 43302.

International Data Systems, Inc. Introduces Altair 8800/IMSAI 8080 Compatible Time of Day Clock

The time of day is now available for Altair 8800, IMSAI 8080, and bus compatible systems from the new IDS model SPM-88 clock board. Current time is read from three consecutive I/O addresses which are selectable in increments of four addresses. No waits are required during clock set or read. This low power board requires less than 400 mA from the +8 V supply and is well below the Altair bus power specification.

The SPM-88 keeps time using the 60 Hz ac line as a reference source (which is very accurate), but provisions are also included for optional 50 Hz line sources, onboard crystal controlled time base, and external battery backup power. The printed circuit board is made of high quality glass epoxy with gold plated edge contacts.

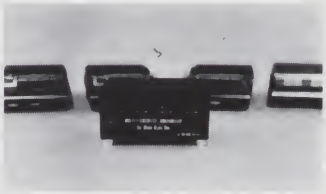
The SPM-88 does not load the CPU in any way except when I/O is executed to set or read the clock. The time is maintained completely on the SPM-88 in contrast to the previously available alternative which requires interruption of the processor and actually keeping time in software. After the SPM-88 is initially set it keeps time even when the CPU is not in run mode, while loading programs, etc.

Software is included with the SPM-88 kit to set and read the clock from MITS BASIC, Processor Technology BASIC5, and from assembler for nonBASIC applications. Software is also included to maintain either calendar of Julian date if desired.

The SPM-88 is available in kit form for $96.00 postpaid and is complete with all necessary components and instructions. Options include IC sockets for all ICs ($10.00), crystal time base ($25.00), 50 Hz operation instead of 60 Hz (no charge, specify SPM-88/50). Delivery is stock to 30 days and orders may be sent directly to International Data Systems, Inc. at Post Office Box 593001, Miami FL 33159.

RO-CHE Multi-Cassette Controller

RO-CHE Systems has just introduced a new Multi-Cassette Controller for the 8080 CPU microprocessors. Their "Magic Black Box" controls up to four cassette recorders with a Tarbell or MITS cassette interface board. This unit handles records, that's right — *records*, to be read or written like a BIG system with four tape drives.



The status bits generated by the interface boards are software controlled to select which recorder to use and, whether to read or write, and to start and stop the cassette. LEDs on the face of the controller indicate which deck has been selected and when that deck is writing.

The software that comes with the unit provides operator instructions to position the tape, put it in read mode, etc. Additional messages also provide for read error recovery. When an input file is opened, the recorder reads to the first file mark of the file and stops. An error message indicates when a tape is positioned past the first record.

To write a record merely load the deck number into the accumulator and call the write subroutine. When the logical output buffer is full, a compressed (no blanks) physical record is written on the cassette tape. The read routine places a physical record from the cassette into the input buffer and replaces the blanks as logical records are used.

The records are written in a format similar to that proposed by Charles H. Eby on page 43 of the June, 1976, issue of *Interface*, titled "Cassette Tape Format Standards," except an interrecord gap is used instead of a byte count. Logical records may be any length.

Using the four port Multi-Cassette Controller ($125.00 in a kit) and $39.95 cassette recorders the microprocessor owner can now have *large* system capability at a small system cost. For complete information, contact RO-CHE Systems, 7101 Mammoth Ave., Van Nuys CA 91405.

Improved Terminal Positive Regulators From National Semiconductor

A new LM140LA series of terminal positive regulators, with several fixed output voltages, in three temperature ranges, is now available from National Semiconductor Corporation. These devices have a 2% output voltage specification, 0.04% per volt line regulation, and a .01% per milliamp load regulation. This usually results in an improvement of two orders of magnitude in effective output impedance, as well as lower quiescent current, when the LM140LA is used as a replacement for a zener diode/resistor combination. In addition, these regulators can be used to provide local on-card regulation, thus eliminating any distribution problems normally associated with single point regulation.

continued on page 14



Meet the Challenger.™



The Challenger Self Portrait

The new price and performance champ from OSI.

He's got his act together!
Even our lowest-cost Challenger comes fully assembled, complete with a 500 ns 6502A, serial interface, 1,024 words of memory and a UL-approved power supply, all for $439. Every Challenger comes ready for easy expansion with an 8-slot mother board, backplane expansion capability, and a power supply heavy enough to handle a full complement of system boards. Our 4K Challenger comes ready to run BASIC minutes after you unpack it. And there's more.

He packs some heavy hardware.
You've never seen memory and interface options like these—not at our prices, fully assembled! 4K RAM memory boards $139! (see below). Single drive OSI Challenger Floppy Disk $990! Dual drive Floppy $1490! Plus 8K PROM boards! A Video Graphics board, including alphabetics, graphics, and **color!** An audio cassette, A/D, D/A and parallel I/O board! A backplane extender board! A prototyping board! And our extraordinary **CPU Expander Board**—it lets you run a Z-80, and 6100 (PDP-8 equivalent) **concurrently** with The Challenger's 6502, or under its control.

There's nothing soft about his software!
OSI has full software support for our Challengers. Including extended BASIC, extended Video Monitor, a Disk Operating System, some very Hollywood real time programs for Video Graphics, Animation, Sound Processing and so forth, plus PROM firmware, with more to come.

He's fast!
You can order The Challenger with a 6502C for a 250 ns cycle time, with a standard 6502A for 500 ns cycle time, or with a 6800 for 1 microsecond cycle time. And with our CPU Expander Board, you can always update to any new CPU to be as fast as fast can be.

And he isn't just good!
He's better! By design. The OSI Challenger is the only completely-assembled, ultra-high-performance, fully-expandable mainframe computer that does this much for this little. Get your hands on one now. Send for your Challenger today.

You can't beat The Challenger!
The OSI Challenger 65-1K. Fully assembled. Features 6502A CPU, serial interface, 1,024 words of memory. **$439.**
The OSI Challenger 65-4K. Same as 65-1K but with 4,096 words of memory. Will run Tiny BASIC without expansion. **$529.**
The OSI Challenger 65V-4K. NO NEED for an expensive terminal. Connects to your ASCII keyboard and video monitor through included OSI 440 Video Board. Features software utility that simulates a deluxe CRT terminal. **$675.**
The OSI Challenger 68-1K. Based on 6800 CPU. For the casual hobbyist, smaller systems. The Challenger 68 series comes only in serial interface forms and is compatible with MIKBug software through an included OSI software utilities package. **$459.**
The OSI Challenger 68-4K. With OSI 4K BASIC on paper tape. **$529**
SPECIAL! ADDITIONAL 4K MEMORY BOARDS. Ordered with your Challenger, limit 3 more at this special Low Price, (total 16K, including 4K already on-board in mainframe). **$139**
Buy 12K or larger Challenger 65 system and we include Extended BASIC FREE!

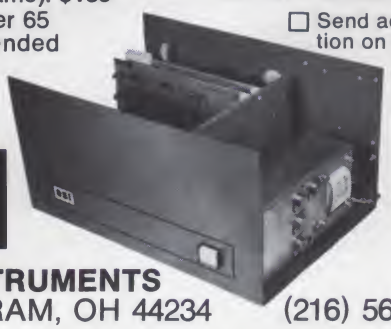
OSI Challenger Floppy Disk System. Fully assembled, for use with OSI Computers only. **$990** Single drive **$1490** Dual drive.
OSI Audio Cassette Interface. Comes assembled, but with room for you to populate with A/D and D/A chips later. (OSI 430 based) **$89**
And all the baseboards and kits of the powerful OSI 400 System.

OK, OSI, I'm ready to buy!
To order your Challenger System, send the total amount of your purchase plus $4.00 for shipping and insurance (plus sales tax for Ohio orders) by personal money order or check. Or indicate **all** numbers on your BankAmericard or Master Charge to charge your order.
Or send a 20% (non-refundable) deposit to receive your order C.O.D. Delivery is typically 60 days (except when payment is by check, which must clear before shipment can be made). Deliveries are scheduled on a first ordered, first shipped basis.

Name_____
Address_____
City_____ State_____ Zip_____
Telephone_____
Bank card info Inter Bank # _____
Expiration Date_____
Account # _____

Check ☐ M. O. ☐ BAC ☐ MC ☐
20%, bal. C.O.D. ☐

☐ Order attached.
☐ Send additional information on The OSI Challenger.
☐ Send additional information on OSI 400 Kits.
☐ $1.00 enclosed for complete OSI Catalog.



OSI

OHIO SCIENTIFIC INSTRUMENTS
Dept.KB 11679 HAYDEN STREET, HIRAM, OH 44234 (216) 569-7945

Letters to the Editor

Can Somebody Out There Help Me?

As a "software" person who hadn't heard of TTL 18 months ago and has little idea of how a regulator or op amp works, I would like to mention some articles on hardware. You said Wayne Green was confused by advanced articles about software in other computer magazines. I am confused by simple articles in *73* about transistors. Don't forget, for every ham operator who knows electronics and wants to program there is a programmer who has always wanted his own computer but does not know how to build those "little things" that always get left out or added on at an exorbitant price (like keyboards and power supplies). You guys know about how those things work – tell us about it and double your audience.

Some examples. Almost every computer kit comes missing the keyboard ($50-200) or power supply ($100-200). So how do you build one or buy one surplus? Hams probably do that as their first project, but programmers don't – they do sorts. I'm building a power supply for my SC/MP. It's the first thing I have ever designed and built (always Heathkits before). I've learned a tremendous amount about the basics which never get mentioned (e.g., how to drill holes in aluminum, where to buy surplus transformers, etc.). I've also learned about using regulator chips; the article by James Kucera in September *73* was very helpful. But I've just begun to learn how little I know.

Here are some simple questions a non-PhD non-ham might have. 1. How does a transformer work? Are there differences I should worry about? How does a high current transformer differ from a low current transformer? Does 12.6 V ac mean peak-to-peak or RMS? 2. How does a diode work? How do they differ? When do you use bridge, full wave, or half wave rectification? What is piv and why do I seem to need 200 to 600 piv for a 12 volt supply? 3. How does a capacitor work? I know that electrolytics are big but what is the difference between small electrolytics, tantalums (more expensive), and discs? When do I use which? How do I choose one, and what should I watch out for if I buy surplus? What are mfd, ufd and WV dc? 4. What are the trade-offs in using a center tapped transformer with full wave vs not using the center tap and using a bridge? This changes the voltage/capacitance needed. What is the cheapest/best way? If I use a surplus capacitor rated for 70 WV dc at 20 V, do I get more farads? 5. The voltage regulators – does it matter which one I use?

More advanced topics. 1. How do you keep noise from motors and things out of the power supply? What commercial power line filters are available? How do I make my own? 2. (A big one!!!) How do pass transistors work? Some designs use PNP, others NPN. What's the difference? How do I find cheap ones? How do I design with them? What is B, Ib, Vceo, etc? A series pass transistor is a lot cheaper than 5-10 regulators. 3. OK, I used a pass transistor, now how do I protect my chips? What is an SCR? How do I use them? What is "fold back over voltage protection"? What kinds of overcurrent and overvoltage protection is available?

Now that I've designed it, how do I build it? How do you mount a T03 transistor? Do you need insulation or heat sink grease. My -12 volt regulator uses the case as input, not ground, what do I do? 2. How do I wire the rectifier? What kind of heat dissipation is required? 3. When do I use a breadboard (I didn't but probably should have)? How do you connect things to it? 3. What about switches and fuses? Amateurs may know about such things when they are born but programmers don't. You switch the ac line, I suppose, but what do you fuse? Will a fuse save the chips? Do regulators ever fail?

Even more advanced topics. 1. Switching supplies, how do they work, how do I build one? Why do I want one? An article on how to build a 10-25 Amp 5 volt switching supply that takes up no space and puts out no heat would be great. 2. Rewinding transformers, is it hard? Surplus transformers with reasonable voltages and currents can be found, but sometimes an extra volt or two would really help.

What if I don't want to design and build myself? Where can I get a good cheap power supply? What OEM supplies are available? What about the Heathkit 7.5 volt 10 Amp supply for $80?

A power supply is basic and so is a keyboard. Altair, IMSAI and SWTPC have power supplies of various qualities but you pay dearly for a keyboard. Where can I get one? There is an unencoded 63-key keyboard available for $20.00, how do I encode it? What should I look out for – no lower case ASCII, not ASCII, etc.? What can I do to change from Hollerith or Baudot to incomplete or complete ASCII? Where have all those cheap Baudot teletypes gone? I have not seen one in many visits to surplus stores. When do I need complete ASCII? When do I need lower case? What about roll-over? What is it? Who cares? (If you need it and you don't have it, roll-over is very frustrating.) What are solid state keyboards? How do they work? Are they worth the money? How do you make a case for a keyboard or a printed circuit board for an unencoded keyboard? What kinds of MOS encoders are available? Which ones have all 128 ASCII characters, which are upper case, which are TTY type? How do you make a TTL encoder? Some sense switch closures while others scan the keyboard. How do I do it?

Hams and programmers alike know the difference between hardware and software. Hardware costs money and can be broken or burned up. Software takes time (which for a "hobbyist" is not money) and is generally non-destructive. A program that doesn't work just needs a little study. You put out hard cash for a burned up 8080. (How about an article on static and MOS devices? How much grounding is required and how do you do it. It's easy to say ground yourself and your workbench but does that mean I have to wear aluminum foil and hold onto a water pipe or what?) All hobbyists have one thing in common, lots of time and little money.

Amateurs have known tricks for years: how to make surplus equipment work; how to buy surplus transformers, capacitors, switches, etc., and save 80%. Programmers like me must learn these things. We aren't rich either and can't afford to spend $200 for a power supply when we could make a better one for $50. But we can't afford to burn up $250 worth of chips either.

So help us, and we can help you. My questions may seem trivial to you, but I know what an assembler, compiler, and interpreter are. More than ½ the members of the San Gabriel Valley Chapter of SCCS are programmers who want to build their own system. Help us in *Kilobaud* with simple hardware articles, and we'll build our systems and write cheap systems software to revolutionize home computing. Working from both directions, we can get there in half the time.

Bill Pearson
Pasadena CA 91125

Wow! Beautiful! What can I say except "Yes, I agree." Potential hardware authors . . . go back and read Bill's letter again. – John.

. . . And Me?

I really enjoyed your articles about computers in the Aug. 76 issue of *73*. I am 14 years old, and before we moved to Texas, I used to be real good at computer programming. The middle school that I used to go to had several terminals connected via telephone to the main computer at Westfield High School. The computer was a Digital PDP/11, BASIC language. The school I went to taught computer programming starting in grade 6. (The name of the school is South Middle School.) When we moved here, I found out that computer programming is not taught until 11th grade! If there is any way that I can use a computer near here, will you please let me know? I really enjoyed computer programming in BASIC language and would love to do it again.

Tom Trusty
2613 Lynnwood Dr.
Arlington TX 76013
1-817-274-7998

Anyone know of a local computer group? Contact Tom. – John.

Mid-America Computer Hobbyists

Congratulations on the introduction of *Kilobaud*. I hope that you hold to your promise to provide introductory and intermediate level construction and software development articles for us neophytes to the world of Hobby Computing. If you can find room in *Kilobaud*, please announce the organization of MACH, Mid-America Computer Hobbyists. Our primary purpose is to exchange ideas and information on construction and software development projects. Interested individuals can contact me at the following address and I will send them information on our next meeting.

Russell W. Steele
MACH
838 Gayle St.
Papillion NE 68046
Attention: Club Secretary

More Reader Requests

I would like to offer a few suggestions as to what I would like to see in the new magazine. First: peripherals should be stressed (this is beyond the usual TVT or video display). A set of peripherals can make a computer into a useful system. For instance, my system has two digital cassettes, a half inch mag tape unit, a paper tape reader and punch, a slowly developing graphics display, as well as the Flexowriter this letter is being written on. We also need to be able to control line powered devices (even if only to impress our friends by turning the house lights on).

Standardization needs to be stressed – both software and hardware. As an example, the 8080 I/O port convention seems to be standard except for the status bits (TBE and DAV). Why so many variations? The Kansas City cassette standard is an excellent idea – the method is terribly slow but it is cheap and uses readily available parts so that everyone can have an interface which will allow him into the software pool.

It is important to generate a software library of low cost (about $5) programs both BASIC and machine language on cassettes and paper tape. This goal can only be met if we have both standardization and peripherals

Letters to the Editor

to do interesting things with. However, any programs offered for sale should have commented source listings, as a large portion of the fun of this hobby lies in modifying things to suit one's fancy.

In any case, you must be careful not to ignore either the software or hardware types. We can all gain by mutual interaction. What all this means is that I hope you and Wayne keep the level and diversity of material in *Kilobaud* the same as in *73*.

Doug Hogg
36 Calle Capistrano
Santa Barbara CA

Needless to say, Doug, we're hoping there are a lot of other people out there with the same ideas you've expressed in your letter . . . because Kilobaud *is definitely going in those directions. And, as far as controlling line-powered devices is concerned . . . keep your eyes open for an article in this issue by Chris Bowick on optoisolators. (We just think we're having fun now! Just wait til we start turning everything on and off with these monsters! I'm not convinced of the practicality of it all . . . but it does sound like fun.) And if you like* 73 *. . . you're gonna love* Kilobaud! *— John.*

Kilobaud — a bright idea

Dear subscription person (be ye he or be ye she)!

Enclosed is a check for an initial sub for Wayne's latest bright idea aka *Kilobaud* in the amount of $12.

F.Y.I.: I am beginning to consider Wayne's "BI's" (bright ideas) as investment material since I sold the first year of my Byte sub for $25 and a duplicate first issue for $15. A total of $40 on a $10 initial investment ain't bad. (Plus Byte is still coming, so far 15 issues on a 12 issue subscription.

Peace be with you and make sure the new mag has a mailing wrapper!

T. Pappan
Corunna MI 48817

Some Do's and Don't's

I'm enclosing $12 for a subscription to *Kilobaud*. Sounds like it will be a good magazine for the tinkerer. There are so many newsletters and magazines around now that I have had to be semiselective about which ones I subscribe to. The main reason I have good feelings about *Kilobaud* is knowing that you are involved and I liked what you and Hal did with the MICRO-8 Newsletter.

I'm looking forward to this new venture of yours.

Let me tell you what I've been doing since I last really talked to you. When we last left Tommy Tinkerer he was fooling with his modified MARK-8. Well, I got turned on to the 8080 before I really even used all the capabilities of my 8008. In other words, I could still be happily tinkering with my Mark-8 except that I guess I am just a hardware freak. When I saw the 8080, I said, "Now there's a processor I could live with for a while." And I especially liked the Altair bus structure. So, I paid $15 for an Altair manual, and proceeded to translate their boards into wire-wrap versions. My experience with the Mark-8 taught me several things, like the value of a "universal bus" structure that allows any board to go in any slot, and also the problems that can be encountered by having the electrons from the power supply regulator travel a long road through several connectors before getting to the board where they are eaten. I was impressed with the Altair's distributed regulation. Anyhow, imitation being the sincerest form of flattery, I decided to flatter MITS by making myself a personal copy of an Altair, executed in wire-wrap.

I had all the parts I needed to do the job, with the exception of the 8080 Chip itself, which at this time was selling at a price of $150 — a bit expensive. However, a microsystem manufacturer (who shall remain nameless) happened to have a certain brand of minicomputer in house, with which I was somewhat familiar. To make a long story short, favors were exchanged, their mini ran a little better, and I was the proud owner of a brand new (free) 8080A.

I had my wire-wrapped Altair finished in about a month. Now, the moment I was dreading, power on . . . no smoke . . . several LEDs lit, but activation of the reset switch quickly told me that my creation was not alive, not responding. I started from the top, clocks were OK, all voltages in spec, front panel one-shots all working right, nothing *obviously* bad. So, I figured the place to start was with the EXAMINE switch, which, as you know, fakes out a JUMP instruction from the front panel, putting a "303" on the bus followed by the settings of the HI and LO address switches. Scoping the bus showed that this sequence was being put on the bus OK, "303 LLL HHH" but the signals looked like they just were not getting pulled up by the data bus pullup resistors. I checked the schematic again. Yes, it said 47k Ohms, and that was exactly what I had. I tried substituting some 20K Ohm jobs, and as I had suspected, the waveform was improved but still not good enough. I didn't want to go any lower on the pullups for fear of causing the 8080 to sink too much current and really die. I spent three weeks on this problem with no solution, I was almost ready to give up when I suddenly got a terrible feeling in my gut, following my last hope, I turned to the parts list in the manual, and there was the value of the pullups: 4.7k Ohms!!! The decimal point didn't show up in my copy of the schematics and it looked like 47k.

One hour and eight 4.7k resistors later, my machine was up and playing music. While I'm talking to you, I'd like to put in a plug for the Digital Group's stuff. I've been using their TVT ever since the Mark-8 days with no problems. In case anyone is turned off by the fact that it has no cursor or that you can't do a "screen read," these are not restrictions as a cursor can be implemented in software by allocating a 256 word screen buffer and using a register to hold the current position. An underscore character can be written at the current position in the screen buffer and each time you input a character, you update the buffer and dump the screen again. Unlike TVTs with a serial interface, a screen dump is almost instantaneous, so this is not as bad as it may sound. I'd also like to say some good things about Dr. Suding's cassette tape interface: it works. Besides the fact that it works, I like the fact that the two frequencies it uses are not harmonics of each other. The system is fast and good and I am using it with a 10 year old cassette recorder that literally traveled in my seabag in my service days and has seen continuous action ever since. By the way, I believe the Digital Group's new TVT offers twice the capacity for just a small increase in price. In short, I'd like to recommend the Digital Group to anyone who is considering buying from them (they also deliver in this lifetime).

For people who are buying components, I have had very good luck with James and also Godbout (and Digi-Key was good).

Poly Paks is generally high priced on most items of interest to micro users, but they have some goodies that aren't carried by others, and they have had a pretty good delivery record with me.

My one outstanding BAD experience was with (who else??) Mini Micro Mart. Despite all the bad things I had heard about them, I ordered a calculator interface from them, along with their "8080 Driver software."

What I finally received was the kit, missing 8 parts (*no* 8080 drivers!) and a note saying that I owed them a dollar for the shipping cost. I sent them back a letter saying that I would be happy to send them their buck as soon as they sent me the missing parts and the 8080 drivers. That was months and months ago, I have received no parts, no drivers, and no acknowledgement of my letter. Since the missing parts were ones which I had in my own stock, I decided to chalk it up to experience and pass my wonderful experience on to others who may be considering buying anything from these people. By the way, the quality of the printed circuit board they sent me was the worst I have ever seen from a hobby computer supplier. It is a two-sided board *without* plated-through holes; and it was not cheap. By contrast, the boards put out by the Digital Group have plated-through holes and are of superior quality.

Tom Boyko
Portland OR 97215



Tom Boyko's Mark 80

Micro-8

Sorry to hear that John Craig had left the Micro-8 Newsletter. After all, if it wasn't for Micro-8 (originally, Mark-8) we wouldn't have had the rapid growth in this crazy computer fever! That little newsletter helped me find my way thru all the confusion when I first started 1½ years ago!

Tate Yoshida
Chicago IL 60616

I've always felt that newsletter did more than anything else to get the ball rolling and I sure enjoyed being a part of it. Keep in mind that it was Hal Singer's idea and most of his effort that made it such a success. — John.

Big Plans

I am already a subscriber to: *73*, *Microtrek*, and *Personal Computing*; now you want *me* to spend *$12* on *Kilobaud!* For anybody else — probably not — but for Wayne Green I'll do it anytime. The *Kilobaud* Quiz in November *73* shows that you have made no little plans for your new magazine. Your emphasis on beginners is important — as even those who claim to be "experienced" are likely to be experienced in only one or two areas and rank beginners in others. Your computer lab should be the *envy* of many a retail store! I hope you will

Letters to the Editor

be smart enough to find some uses for all that computational power in the business of publishing your two *great* magazines and many books. By the way — will you be providing tours of your lab for those fortunate few that are able to find their way to the great metropolis of Peterborough?

Your RTTY types ought to be the ones most into microcomputers — followed by CW (the human voice has got to be the *worst* tool for data entry into a stupid machine — followed only by handwriting). As a RTTY type — you might think of some ways that hams and microprocessor enthusiasts might join together for some valuable public service activities — both of an emergency and nonemergency nature. Give your readers some ideas and perhaps some of the application programmers among us can come up with some software to match. Keep up the good work and — never say die.

Kenyon F. Karl
Waterville ME 04901

Okay Kenyon — programmers please note — and visitors? You bet! Love 'em . . . Wayne.

Sharing — The Lifeblood of a Hobby

During the past several months I have become heavily involved in the hobbyists computer movement and have had occasion to read numerous articles, both for and against the sale of software to hobbyists by various companies. There also appears to be great concern over the "piracy" of this software by those hobbyists who either cannot afford or cannot justify the purchase of something as intangible as a computer program. I would like to take this time to voice my own personal views on this subject and also to introduce to you a new concept regarding the distribution of some software which I have developed.

My experiences in various hobbies range back over twenty years to my involvement in ham radio experimenting. Although I have been out of touch with this market for many years, I seem to recall a genuine interest within the group of ham radio enthusiasts in sharing all their knowledge and experiences with others involved enough to take the time to listen. I feel quite certain that it was this high energy level and genuine interest to share with others that played a great part in promoting the hobby to the level it is today and also in the various technical developments that came about because of the acceptance of the hobby. At no time can I recall similar instances of concern over the piracy of ideas or schematic diagrams which seems to loom like a huge black cloud over the computer hobby market of today. Ham radio operators were all too enthusiastic about sharing their developments to become entangled in the web of legalities that enshrouds the patent infringement laws. It seems to me that the few unscrupulous individuals that may have profited from someone else's ideas in the ham radio field cannot possibly overshadow the opposing mass of those who gained knowledge and pleasure from the sharing that seemed to prevail.

My point in this matter is simple; I feel that if this is to be a hobby that is to be accepted by the general public, it should be promoted in a similar manner to the tried-and-true tradition of ham radio. The sharing of ideas with others should be encouraged and the profit-making in large volume should be left to the commercial data processing market which still appears to be totally viable, judging from the massive amounts of commercially available programming talents. I, myself, have been a computer consultant to various large firms for over six years and have never felt the need to charge for programs that I have developed solely for the amusement or education of those people not in the commercial market. I am a firm believer in this theory that any energy you expend to help others will eventually come back to you in some form or another. The times I have taken to assist students in their programming endeavors has always seemed to provide me with new viewpoints on my own ideas and has never threatened in any way my ability to make a living as a professional programmer. The vast amounts of talent, that is yet untouched, in the new breed of computer enthusiasts should not be stifled because of lack of assistance from those of us who have been in the professional end of the field. During the past several months I have had the opportunity to "hang around" and perhaps at times make a pest of myself at my local computer hobby store, The Computer Mart, in Orange, California. I have viewed, first hand, the enthusiasm that prevails over the range of people buying these fancy machines and have been amazed at the innovativeness of some of them to solve problems which in all practical respects should have no logical solution. These people in the end are going to provide us with some genuine technology that can do nothing but be beneficial to everyone and I feel that they deserve all the assistance they can get from the professional community.

Fortunately, the people who run The Computer Mart share in the above views and opinions and have assisted me in developing a system which I feel will be beneficial to those who get involved with it. I have developed a 16-bit microprocessor system which will plug directly into existing IMSAI and ALTAIR computer systems and use all the existing 8-bit memory and peripheral gear with little or no modification. The software that will be offered is a modified version of a complete commercial operating system that I have developed over a period of four years and have sold to the various commercial users for prices that range into five figures. It includes a flexible disk-based executive with timesharing capabilities and modular construction which will allow the hobbyist to custom tailor it to his order. The I/O is file-structured and multi-user based with easy to understand subroutines to allow new and wonderful I/O devices to be added to the system. The system also includes a macro-level linking assembler and linking loader, an 8080 cross-assembler and loader, character string text editor, monitor generation package, and all utilities necessary to maintain the file-structure data base. I am currently in the process of developing a powerful BASIC language which will be both interpretive and compiler oriented and should include most of the popular features currently found in the extended BASIC packages. This is the first piece of software of major proportions that I have developed which was not under contract to some commercial user and I find it quite relaxing not to have to work under a set of predefined constraints.

I intend to distribute this entire package through the Computer Mart for a price that will cover the copying media costs and labor involved to do the distribution. The tentative price we have set for the software system is around $50.00 for the entire package, including BASIC which will run under the executive or as a free-standing program. I also intend to make all source files available to anyone who wants them for understanding or custom modifications. The source files will be available typically on several floppy disks, in a format that is easily assembled by the system assembler, and on microfiche listings. The documentation in the form of comments within the source files is second to none and makes for easy understanding by other programmers and also by myself. (How many times have you gone back to an old program and asked yourself "Now why did I ever use that register for that routine?" and puzzled over the answer for several hours?) Additional external documentation will be available to assist the hobbyists in customizing the system to his own personal needs.

Another service I intend to provide is a personal response to all questions and comments concerning the system or bugs detected (and I'm sure there will be quite a few) in as quick a manner as I can arrange. A self-addressed stamped envelope will be the total cost of this service, which should be within the grasp of even the most insolvent enthusiast.

I honestly feel that this type of attitude is what is desperately needed in the computer hobby field to keep us all in the current state of high interest and development. Perhaps no one will get rich quick by selling software for low prices, but I feel there are enough other areas to be pursued in the commercial arena without penalizing the student who merely wants to learn about computers and figure out a niftier way to catch those elusive Klingons!

Dick Wilcox
Tustin CA

You sure have some refreshing ideas, Dick. I've seen a number of significant contributions made by professional hardware and software people and it's nice to know that the "sharing" philosophy is still alive and well.

We'll certainly be looking forward to your article describing that 16-bit system in an upcoming issue of Kilobaud. *— John.*

6100 Anyone?

I would like to see some detailed discussions and/or comments on the Intersil 6100 microprocessor (PDP-8 equivalent) as featured in Ohio Scientific Instruments 400 system (configured as a PDP-8 emulator). I'm going to build a PDP-8 emulator machine in the near future, and would like to see comments on OSI's complete 400 systems — they look good (to me, anyway — inexpensive). Software for the PDP-8 is easily purchased, and most important, there is an ocean of inexpensive PDP-8 software available.

James E. Fletcher
Parkersburg WV 26101

There are probably a lot of people out there who would like to hear about systems built around the 6100 (whether it's OSI's, or someone else's). If you're building one (yes, you . . . out there) then how about keeping notes on the problems and successes you encounter with the thing and do a write-up on it? And, is this emulation of the PDP-8 all it's cracked up to be? Who has built a 6100 system and actually taken PDP-8 programs (on paper tape, for example), loaded them into the machine, and run them? I, and a lot of other folks (like Jim up there), would like to hear about it. Okay? — John.

Kilobaud — An Interface

I have recently received two issues of *73 Magazine* and found both of them very good, informative and at a level I could understand. If *Kilobaud* follows this same format, it will be an excellent magazine.

I'm interested in radio-teletype from a computer hobbyist standpoint. The computer system I'm currently trying to set up and run is an Altair 8800 with an SWTPC TVT-II, PTC 3P+S interface board, 4K memory, PerCom CI-810 cassette interface, surplus printer and other odds and ends. I need a magazine such as *Kilobaud* to help me interface every-

continued on page 67

BOOKS

**The Compleat Computer,
Dennie Van Tassel,
Science Research Associates,
1540 Page Mill Rd.,
Palo Alto, CA, 1976,
$5.95, 9"x12", 216 pages.**

This nicely produced book is a tasty collection of news articles, cartoons, fiction, poetry, full color reprints of early science fiction magazine covers, thought provoking articles, and extensive references to other literature. The carefully conceived goal is to get you thinking about the Big Picture — how computers fit into our society and our day-to-day lives.

The Chapter titles give a feeling for the coverage:

In the Beginning; How Computers Do It; The Software; The Present and Potential; Applications; Governmental Uses; The Impact; Controls, or Maybe Lack of Controls; Your Future.

Put this book on a coffee table, bedside table, or any "heavy traffic" area so people can grab it and read it in little spurts. It's informative in an entertaining way, and people will keep coming back for more.

Rich Didday
Santa Cruz CA 95062

**WRITING AT WORK:
DOs, DON'Ts, and HOW TOs
By Ernst Jacobi
Published by Hayden Book Co., Inc.
Price: $6.85
198 pages, softbound**

Don't you want to make extra bucks to support your computer? Sure you do! Don't you have discoveries to share with Phellow Computer Phreaques? You bet you have! Then what's holding you back from writing an article that will earn money? Is it that you're just a bit timid about putting your thoughts on paper for *everyone* to see? We understand. So we've found help for those of you who are teetering on the brink of imparting your stored-up knowledge. It's a book that tells in easy-to-understand, step-by-step terms how to write articles, news releases, product reports, and even interesting interoffice memos. Our interest of course is the help it offers for magazine article writing.

At the outset, Jacobi states that you, as a potential writer, probably already have a good command of the language, grammar, syntax, and vocabulary. He assumes that you have something important to say. His goal is to "turn writers into communicators." From the book's preface, his purpose: "Form without substance is meaningless. Substance without form creates needless difficulties. But if there must be a choice, there is no question but that in the long run the world will choose substance over form, interest over mere clarity. For whatever else you may have to offer as added inducements, to be effective your communication has to be interesting — first and last. This is the major point I make in this book."

Jacobi also discusses essential components of a well written piece by offering examples of extracted writings for comparisons of good and bad form. The extractions are, in themselves, interesting reading. We are given tricks that make it all seem so simple; the first, "Write it as you'd write a letter to a friend." (Which would certainly apply to writing for *Kilobaud*.) Finding that to be an aid on an article we were currently writing, I checked our article continually against each new point the book offered. Each point was well stated and exemplified, making quite clear to me that there is certainly room for improvement in my own work.

The first section of "DOs" includes "Woo your Reader", "Have a Point of View", "Select your Point of Attack", and 15 other topics covered in 73 pages. All are interesting to read and were immediately helpful in my writing.

The "DON'Ts" section dashes myths about writing that have been held dear for too long. For instance, "Don't use surprise endings", which are great for novels and short stories, but have a negative effect on informative articles. Don't use passive voice in your writing. The author encourages us to be informal (using words like "I") and demands that "I" take direct, but comfortable, responsibility for my words while talking to "you." I especially applaud Mr. Jacobi's discussion of the use of cliches and, without reiterating his arguments against their use, simply restate his advice: "Abandon them."

Perhaps the section entitled "HOW TOs" is among the most valuable. "How to Persuade", "Why and How to Punctuate", "How to handle Statistics — Tables and Charts" and many more topics fall into categories covering the finer points of writing for publications.

Ernst Jacobi uses a chatty and personable style, making this guide to communicative writing his own case in point. (Was that a cliche?) He follows his own advice by making the book lively, informative, and easy to read. Jacobi's credentials include 25 years' experience with major corporations as a writer and editor, and as an instructor in business communications. Now that I've found it, the chances are I'll keep this book at my right elbow whenever I write an article.

Sheila Clarke
Glendale CA 91206

**A Computer Perspective,
(The Office of) Charles and
Ray Eames, Harvard
University Press, 1973,
$15, 9"x9", 175 pages.**

This extended photo essay makes the history of computing come alive. Every page is covered with photographs of founding fathers and mothers, their inventions, their private notes and letters.

The coverage begins with Charles Babbage and his "Great Calculating Engine" and runs chronologically up to the delivery of the first commercially available digital computer, the UNIVAC, in 1951. Along the way are literally hundreds of fascinating tidbits, bizarre machines, the fossil traces of an incredible technological evolution.

For me, the snapshot (p. 133) of Goldstine and Eckert, beaming like proud papas as they hold a memory unit from the ENIAC, is worth the price of the book itself. The thing is wider than the two of them, looks to be about a foot high — and stored *one* decimal digit!

I found this book utterly fascinating. Not only couldn't I put it down until I'd finished it, but after two weeks I still found myself flipping back through it, amazed.

Rich Didday

**8080 Programming For Logic Design
Adam Osborne and Associates, Inc.
P.O. Box 2036, Berkeley CA 94702
$7.50 (paperback)**

In the commercial/industrial world we find that engineers are quite often using microprocessors for entirely different purposes than the owners of personal computer systems. These engineers are usually putting the micro to work in "dedicated controller" applications and/or as a substitute for logic circuitry. Some of their techniques and approaches to problems will undoubtedly be of some help to hobbyists doing similar design work. In the years (and months) to come we're going to see more and more hobbyists developing interfaces using microprocessors instead of logic elements.

The premise of this book, which it shares with some other recent volumes, is that special purpose assemblies of logic elements can be replaced by general purpose microcomputers with appropriate software and some minor amount of special purpose hardware. This particular book deals only with the 8080 processor but it does so in considerable depth and with a very detailed presentation of basic concepts using illustrative examples.

The first topic deals with the means by which a microcomputer can perform the functions of individual logic elements. This discussion starts with a description of how the 8080 can simulate the operation of a simple logic inverter and proceeds through consideration of progressively more complex elements: a variety of gates, different types of flip-flops, one-shots, etc. Each element is analyzed in extensive detail, and the reader is given some appreciation of both the hardware and software aspects of the particular 8080 processes involved in simulating that specific logic device. Numerous illustrations are a great help in following the text.

The next section considers the application of these logic processes to a specific example, the print wheel control logic for an actual printer. First the operational requirements of the control logic are defined and the implementation of the control logic with conventional IC logic elements is presented. This logic assembly is then converted to 8080 implementation on a one-for-one basis, the computer performing a specific operation (or operations) to replace each logic element. This is clearly an unrealistic approach to design, no matter how instructive. The following chapter acknowledges this fact and starts over, taking a different approach. Beginning again with the operational specification and the necessary input and output signals, the print wheel logic is redesigned, considering only the best use of the microcomputer capability and ignoring the implementation with discrete logic elements. These chapters are most effective. In going through the complete design process from three different approaches the reader comes to a solid understanding of the problem and is therefore better able to appreciate the solutions.

The next chapters deal exclusively with software. First the print wheel control software derived in the preceding chapter is examined. Some more sophisticated programming techniques are introduced and the effectiveness of these techniques is demonstrated by improvements in the print wheel control program. The concluding chapters examine the 8080 instruction set and some commonly required subroutines. Like the rest of the book, these chapters are exhaustively thorough and generously illustrated. These chapters would be useful by themselves as instructional and reference material.

In general, this is an excellent book for the reader who is willing to dig very deeply into the subject. It is painstakingly thorough and the reader willing to take those pains should end up with a solid understanding of many useful concepts and techniques.

There are two objections to the book. The first and most obvious is the format of the printing. The author uses boldface and lightface type to distraction. In theory the lightface type only expands on the material printed in boldface. As a practical matter, however, the reader is advised to read all of the text. It will require more time and greater concentration but this will be repaid by better understanding of the material and therefore greater retention. In addition, it will put to rest the nagging suspicion that you're missing something important in all those paragraphs of lightface type.

The second objection is the use of only a single application example. Having established a very firm foundation with the analysis of the print

continued on page 19

LOOKING AHEAD

Rich Didday
1218 Broadway
Santa Cruz CA 95062

Rich's "Lookahead" forum is an invitation to you, the reader, to submit material ranging from letters to news clippings to one page articles on the impact personal computer systems (and related items) are going to have on society. Aside from doing some fine writing for 73 Magazine *in the past (you really should catch his three-part series on BASIC programming techniques), he also writes for Matrix Publishing. Be on the lookout for his new book, entitled "128 Q & A About Home Computers."*

[I really must share a neat story about Rich with you (I'm sure there's a good moral here, too). When Rich finally made his decision as to which home computer system he wanted, he sat down and wrote the check, mailed it off, and then began the long wait. And he waited . . . and waited . . . and waited. Eventually he got around to writing a couple of friendly reminders, but he finally had to resort to a very *nasty letter. The letter worked, and a few days later he received a refund check in full. Now, here's the good part . . . Matrix Publishing agreed to fix him up with a computer (and it turned out to be the same one he had been trying to buy). Sure enough, he received the computer from the company he had been threatening the same day he got his refund check from them!] — John.*

WHAT'S GOING ON HERE?

The home computer idea, movement, revolution, avalanche is upon us. It's getting bigger day by day, hour by hour, microsecond by microsecond. It's the modern media world of microminiaturized, hypercybernated, kilobaud, megabyte, electronic wizardry, and you don't want to be found wanting when the computerized future comes knock, knock, knocking on your door. The future is here now! In fact, you just missed it! Your skill, your equipment, and your psyche are all out of date. Catch up! Get new equipment! The information explosion will inundate you if you don't get on the stick right now. Hurry, hurry, hurry! The onrush of progress is expanding exponentially, so get moving *now!*

But, on the other hand . . . if you can stand aside quietly for just a little while and keep your head (perhaps chanting "But What Am I Going To Do With It?"), pretty soon your heartbeat and breathing will return to normal, and maybe you can start to see it all as a colorful carnival, thrashing and flailing its own merry way into the future. Can't you just see the advance men, the barkers, the promo crew making up leaflets and throwing them to the crowd on the run? And look! There-a clown dressed up as a robot has stopped to give an IC chip to a little girl! But there's a more ominous note too. Can you spot the technological storm troopers, the computer shock forces, the agents of change. The carnival is sending little jolts through the crowd as it passes. As it pulls out of sight and the sound of the drums gets softer and softer, the crowd starts to break up, somehow a little different than when they came, wondering what has happened.

But on the third hand (bet you wish you had one), you wouldn't be reading this, and I wouldn't be writing it if we weren't in both places at once; both in the carnival *and* in the crowd — enthused and fascinated by computing, but, at the same time, members of a society that is being changed, perhaps in ways we don't understand, by what we're doing.

Our society *is* going to be different because of what we do. But *how* is it going to be different? Are there alternatives that we, as a society, should be talking about and consciously considering before it's too late? Is there any hope that we can do something meaningful, useful? There must be a few clues around that we can uncover. After all, this isn't the first time a new, broadly applicable, radically different tool has started to catch on. It's certainly not the first time that a new technology has infiltrated an unsuspecting society.

But then, maybe the reason our society has been so unsuccessful at anticipating the real consequences of new technologies is that it's just too difficult. Imagine standing beside the assembly line in Highland Park, Michigan in 1914 as the first Model T came chugging off the assembly line. By what sort of clairvoyance could you have imagined interstate highways, drive-in movies, shopping centers, teenagers making out in back seats, suburbs, smog, *Los Angeles?*

But wait. Maybe if you were careful, kept asking yourself "What if everybody did it?", watched for trends, watched what people actually did with their cars, talked to city planners, car builders, and owners, then maybe, just maybe, you would have been able to piece together what was happening. At least enough to put up a respectable fight at a party when someone started babbling about how nice it would be to get the streetcars out of the city because it was so unpleasant to drive over the tracks.

This isn't the first time that an advance in electronics has spawned a new species of smaller, cheaper digital computers. Computer "revolutions" have swept through the corporate and governmental subsocieties at least three times before (the first generation, IBM 360's, minicomputers). Surely we can learn something from those experiences.

So there is hope. If we act now, we can become a sort of distant early warning system. If we alert ourselves, sensitize ourselves to the impending changes, and share our observations with one another, we'll at least be able to watch things unfold. And maybe we'll be able to see warnings of



Hurry! Hurry! Hurry!
Step right up!
Get your future now!

unwanted changes . . . before it's too late.

If you want to get involved, what can you do? Lots. First of all, let's make this forum serve as a communications medium, make it work for all of us. Send in your ideas, suggestions, complaints. Send in a news clipping, a note, or even a whole magazine page on a subject of interest to you . . . and the readers.

If you have some special expertise or interest, use it to give us a report from that point of view. For instance, if you have a background in economics or budget analysis, you could work out some rough estimates of the impact of home computers on the economy under a range of assumptions — what if they become as popular as microwave ovens . . . stereo equipment . . . television?

If you are on one side (or the other) of the law enforcement business, let us know of any new ways organized crime is using computer technology. And of course, there's the whole invasion of privacy can of worms.

If you work for a company that uses or makes computer equipment, spend a few lunch hours talking with people from different departments. We need to know what management, marketing, production, and research people think, feel, and plan to do about computing. Surely each group has a different perspective.

If you are on friendly terms with your local school, why not conduct a survey? Find out how student attitudes about computers, calculators, electronic games, and media differ from those of teachers and parents. Could it be that children really are learning more from TV than the classroom? Can you test that idea?

If questions about the societal effects of home computers come up in a bull session, jot down the various points made, and send them in. Let us share your ideas. Don't let your brainstorms be forgotten.

If you watch a lot of TV, keep a pad of paper near by and jot down references to computers, robots, electronic technology, etc. A well done TV diary, giving show, time, incident, and your ongoing analysis of the attitudes and assumptions revealed in each instance, would make a great Lookahead contribution.

If you like to hang out in pool halls, bars, and penny arcades, see if you can get the management to give you figures on which games are getting the most plays? Try to get a feeling for what draws people to specific games. Play a number of games on the old style machines and then on the newer computer driven ones. Tell us how you see the difference. How do you feel after winning or losing at Quick Draw or Jaws?

How about a collection of computer oriented graffiti! It seems that the most dynamic, active, and influential companies, colleges, and computer centers are the ones with the best bathroom wall philosophers. And graffiti seems to be a real outlet for people's feelings and fears. "Why look up here? The joke is in your computer."

If you read a lot of science fiction, keep us abreast of new ideas there. A side note: It seems to me that virtually all sci-fi treatments of computers to date have been off base. They all suffer from the monolithic, centralized super-computer madness of the sixties. Isn't it obvious that what we're facing today is a society which has an abundance of computer power distributed widely throughout it rather than centralized in one god-like super-computer? Seems to me that a more reasonable model of computing in the future is the anthill or the beehive, not Jehovah.

One note of moderation: I think we should stay away from trying to guess what specific pieces of hardware are going to be available in the next few years. The manufacturers will do their best to keep us well informed of that.

After we've seen a number of changes being brought about by the infusion of computer power into our day-to-day lives, we'll be able to construct a number of different plausible futures and different scenarios. (And of course, we'll want to keep close tabs on how we and our friends are learning to use home computers by reading the rest of the magazine!) Then we'll be in a position to put new developments in perspective and to suggest alternatives. But first we need the raw data, a collection of facts and observations from different points of view. Let's get started!

Due to the various time delays in the system, letters you write now probably won't show up here until two issues from now. But please keep sending your thoughts in, even if to say that you don't agree with my proposal.

Next month, I'll discuss some past predictions that have been advanced. The majority of them have turned out to be wrong, and it's interesting to speculate on where they went astray.

Meanwhile, let's keep our feet on the ground, and our eyes and ears wide open.

Send correspondence to:
Looking Ahead
1218 Broadway
Santa Cruz CA 95062

IN WIRE-WRAPPING ok HAS THE LINE...

HOBBY-WRAP-30 FOR AWG 30 WIRE ON (.025 SQUARE POST)



STRIP WRAP UNWRAP

OK MACHINE & TOOL CORPORATION
3455 CONNER STREET, BRONX, NEW YORK, N.Y. 10475 U.S.A. • PHONE (212) 994-6600
TELEX: 125091 TELEX: 232395

NEWS OF THE INDUSTRY

from page 6

The available output voltages, which range from 5 to 24 volts, allow the LM140LA series to be utilized for a wide range of solid state equipment applications, including logic systems, instrumentation, and hifi. Although they are primarily intended to be used as fixed voltage regulators, these devices can also be used with external components to obtain adjustable voltages and currents.

With adequate heat sinking, the regulator can deliver up to 100 milliamps of output current, but the inclusion of current limiting ensures the peak output current will remain at a safe level. Safe area protection for the output transistor is another feature which prevents the IC from overheating. This limits internal power dissipation and will cause a thermal shutdown circuit to take over if power dissipation exceeds the level allowable by the heat sinking provided.

The LM140LA, LM240LA and LM340LA are all available in the low profile metal T0-39 package, and the LM240LA and LM340LA are also available in plastic T0-92 packages. When ordered in lots of 100, the price of the commercial version is $2.10. These devices are immediately available from stock. For more information, contact Dave Whetstone, National Semiconductor, 2900 Semiconductor Dr., Santa Clara CA 95051. Phone: (408) 737-5000.

Make Your Own Computer-Controlled Plotting Device!

Sylvanhills is now offering do-it-yourself X-Y plotter that may be of interest to the computer hobbyist. As shown in the photo, the mechanical system comes completely assembled, aligned, and tested. The electronics are supplied in kit form. The electronics consist of two motor controllers and one pen control. The inputs to the controllers are latched and require only one TTL U.L. so the system can be hooked directly to the I/O port of any microprocessor.



While the interface electronics assembly is straightforward and can easily be accomplished in two to three hours, this kit is not recommended for the beginner. Also, since the kit contains no software other than assembly instructions and some basic suggestions, the user will need considerable expertise in programming.

The kits range between $750 and $895. Ancillary equipment is also available. For more information, please contact Allen Penn, President, Sylvanhills Laboratory, Inc., 1 Sylvanway, Box 239, Strafford MO 65757. Phone: (417) 736-2664.

Sylvanhills Laboratory TTLP Logic Probe

Microcomputer hobbyists in need of a first test instrument will undoubtedly consider acquiring a logic probe. Most probes are in the $50 plus price range, but not the TTLP. For less than $30 (with options), the Sylvanhills model (originally designed as a classroom aid) will indicate actual 1s and 0s through a seven segment LED display.

The TTLP also features storage of fast negative-going signals. The whole package is mounted on an epoxy G-10 double-sided, plated-through, printed circuit board. The TTLP has a sharp point for digging into nodes, making it easy to penetrate corrosion. Logic 0 thresholds are from 0 to 0.8 V, Logic 1 thresholds from 2.2 to 5 V. Maximum voltage on the standard version is 5 V, with an option available to provide overvoltage protection up to + 25 volts.

It is important however, to keep the TTLP's limitations in perspective. Since a clock pulse alternates between 0 and 1 on the seven segment LED, it appears as a zero with a low intensity. If the signal is a low-going high pulse, the zero indication on the LED will not dissipate fast enough to show the pulse. One way to get around this deficiency is to check the signal through an inverter.

The prime selling point of the TTLP is the fact it is inexpensive. The device will give you an indication of what's happening in your system, although it lacks all the functions of a full logic probe. For more information, contact Model TTLP Logic Probe, Sylvanhills Lab, Inc., Sylvanway, Box 239, Strafford MO 65756.

TYCON's 8080 Development Software

New software development packages for 8080 type microcomputers (and priced for the hobbyist) are now available from Tychon. Tychon's Editor (TED), Assembler (TAS), and D-BUG programs can be run in microcomputers with at least 4K of read/write memory. Programs are available on paper tape or in 1702A or 2708 type PROMs. Each software package includes complete documentation on its use and information about changes for different I/O formats. Listings for each program are also available.

Prices for the software are as follows: Editor/Assembler tape — $25; Editor/Assembler listing — $40; D-BUG tape — $10; D-BUG listing — $40; Documentation package — $5; and delivery is within 10 days. For further information, contact J. A. Titus or G. H. Wilson at Tychon, Inc., PO Box 242, Blacksburg VA 24060. Phone: (703)-951-9030.

New Video Game Chip From National

National Semiconductor Consumer Products Division has received FCC approval for its TV game, "Adversary" which employs the MM57100 video game logic circuit and LM1889 video modulator.

"Adversary" (suggested retail — $99) features a choice of three playing fields: tennis, ice hockey, or handball.

For more information, contact Georgene H. Berglund, Public Relations, Consumer Products Division, 1177 Kern Ave., Sunnyvale CA 94086.■

```
1 Print"Publisher's Remarks"
10 Do End
Run
```

from page 3

was getting $10.20 after royalty to the author. Subtract another $1.20 for the manufacturer's representative who sells the programs to the computer stores, sets up the displays, follows up on problems and collections, and makes sure that the stores have a good stock of the latest programs as well as that the salesmen understand what the programs will or won't do.

Each program has to be checked and rechecked before publication, the documentation has to be written and published, the sales package has to be designed and printed. Computer tape is not cheap — it will probably run around $4 a cassette for tested tape. Once duplicated (have you seen what top quality duplicating machines cost?) each tape will have to be checked by a computer to make sure every last bit is okay. Then comes the advertising and sales campaign — inventory — shipping — billing — collections — sales managers — the works — plus overhead, and even a modicum of profit.

So much for programs . . . I got carried away.

I'm hoping that we'll be able to have enough programs and info in *Kilobaud* so there will be no need for user groups of the various systems.

There is one more major thrust for *Kilobaud*, and this has to do with that small business market I mentioned before. This is a categorization, really, for it also includes home computers and school computers . . . all of the uses to which we expect small computer systems to be put. I figure that this is going to work out at least as well as the projections made for it — perhaps 5000 computer stores each doing an average of $500,000 per year — some larger, many smaller. I'll be surprised if most of these businesses aren't run by ex-hobbyists. In *Kilobaud* I'll try to help all I can not only to make the computer hobby fun but also to show how it can be used to make money. One little hint that works for you will be worth many times a Lifetime Subscription cost.

So there are my aims — we'll see what happens — and please remember that *Kilobaud* is a medium for you to communicate with other hobbyists. The more we help each other, the better off everyone will be.

One commercial factor — it is the advertising that pays for a magazine to be published, not the subscriptions or counter sales. This means that the more you let advertisers know that you are getting their message, the bigger and better your magazine will be. Send for their literature and let them know when you buy something that *Kilobaud* helped the sale. If you do this you'll have as fat a magazine as the hams have in *73* and your main problem will be finding enough time to read all of the articles.

Hopefully you'll be taking enough advantage of the money-making schemes in *Kilobaud* to buy all of the equipment you have time to use — or at least you'll be writing articles which will get you money for hardware. You have an opportunity unparalleled in history to make money from your hobby — don't ignore that insistent knocking at your door.■

The BASIC Forum

Dick Whipple – John Arnold
305 Clemson Drive
Tyler TX 75701

Recognizing the popularity of BASIC as a programming language among computer hobbyists, the editor of *Kilobaud* suggested that we create The BASIC Forum. As the name implies, our effort will be to provide a medium for the exchange of ideas on BASIC. As authors, our responsibility will be mainly to compile and edit information provided by readers and other sources. Publication of The BASIC Forum on a continuing basis will depend, to a great extent, on the interest and response shown by readers.

Operation of The BASIC Forum is best explained by the diagram of Fig. 1. At point A, the authors (Dick Whipple and John Arnold) will receive input data from *Kilobaud* readers, commercial software suppliers, and any other good sources which develop. We will edit and condense the material for publication in *Kilobaud* under the title "BASIC FORUM" (point B of Fig. 1). *Kilobaud* will then move to the readers via link C. The readers at point D will then have an opportunity to consider the various ideas put forth. From the readers, two potential paths can develop: (1) from reader to reader (Path E), and (2) from reader via link F back to the authors at point A. In this way, two continuous information loops will be established. If no glitches spoil the system, a significant idea exchange is possible.

To bring the system up from a "cold start," may we suggest some possible reader input. This may consist of one or more of the following data types:

1. Questions about BASIC and about programs written in BASIC.
2. Comments and suggestions regarding the use of BASIC in applications involving business, industry, education and recreation (games). *Note:* The order is not significant!
3. Information concerning the availability of BASIC system software and application programs.
4. Reports of problems and errors encountered in the use of BASIC software and programs.
5. Requests for assistance (or offers to help) in developing specific applications programs (part of the reader-to-reader loop).

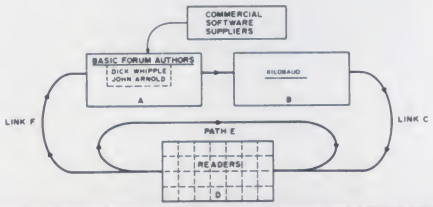


Fig. 1
Flowchart
The BASIC Forum

Upon receipt of this data, the authors will sort and organize the material for presentation in future issues of *Kilobaud*. Every effort will be made to fairly evaluate the data and we will try to publish as much as practicable. From time to time we will introduce topics concerning BASIC to help stimulate reader interest.

To get the system initialized and off to a good start, we submit the following topic for your consideration: The need for BASIC language standardization has been under serious discussion lately. Even now efforts are being made to bring the same standardization to BASIC that exists with FORTRAN and other high level languages. *To what extent (if any) does imposing such standardization influence creativity in the programming field?* There is little doubt that such standardization produces better program exchange potential, but is this at the expense of experimentation and diversity in programming? Let's get your ideas!!

Material for BASIC FORUM should be addressed to:

The BASIC Forum
c/o Dick Whipple
305 Clemson Drive
Tyler TX 75701

If you wish to have any material returned, please include a stamped, self-addressed envelope. All items published will be properly acknowledged according to source. If you desire to enter the reader-to-reader loop, let us know and we'll publish your address as well.

Our RESET switch has been thrown — we are now entering a data entry mode. Start your data flow! ■



Around the Industry

Wayne Green

THE HOBBY COMPUTER INDUSTRY

If you read many of the professional computer magazines, you probably realize how small the hobby part of the microcomputer industry is as compared to the professional end – and the two are quite separate in many ways.

In order to keep in touch with what is really happening in our microcosm I've been getting around to visit most of the major firms supplying equipment to the hobbyists; and I thought you might like to know what I've been able to discover.

My first trip around was in August 1975 when I visited Sphere, MITS and Southwest Tech. These were all there were to visit at that time. I talked quite a length with the three and reported this trip in *73* (Feb. 76, page 86). It seemed appropriate to make another go-around in August 1976 for a follow-up on Sphere, MITS and Southwest Tech, plus a visit to newcomer firms such as Jolt, Apple, M&R, Imsai, Wave Mate, Morrow, Godbout, and Intelligent Systems.

JOLT

Though the Jolt system has not been advertised much recently, it was one of the first 6502 based systems and, from what I had seen at hamfest demonstrations, was a very cleverly designed microcomputer. We (Sherry, the marketing manager and I) dropped by to talk with Ray Holt and Manny Lemas. Ray is the engineering end of the business and he showed us the Jolt computer and their plant.

The plant is a small one – most of it in one large room, and I didn't see any signs of high volume business, so I suspect that the lack of advertising has slowed things down. Ray showed us the Jolt system, including his new 8080A CPU and video tape controller. The designs look very good, but I suspect that there probably has been a good deal of resistance to the lack of a cabinet and to the mechanical design, which is more reminiscent of laboratory equipment than a computer. The fact is, as more and more manufacturers are finding out, the design of the cabinet and front panel are of critical importance in sales – this is show biz more than engineering.



The Jolt 6502 CPU kit.



The Jolt work area where kits are packed for shipping.

The Jolt kits are very nicely packaged on foam sheets. The boards are top quality. They have the CPU card, a 4K RAM card, an I/O card, a power supply card, and a 1K assembler which comes on PROMs. The video tape controller system keeps track of any place on a tape and returns to it at the touch of a button. It also responds to answers a student provides to questions on video tape and returns the tape to a programmed frame if the answer is not right. It appears to be a great compliment to video tape teaching systems.

Much of Ray's and Manny's efforts are going into their Microcomputer Digest, a monthly newsletter aimed at the professional side of the microcomputer industry. At $28 per year this newsletter is not very attractive to hobbyists, but should find acceptance within the OEM part of the industry.

Now, about that mechanical design – the individual boards are stacked one on the other with metal spacers in the four corners to hold them apart. It makes a compact square computer, but I can see where servicing could be frustrating for there is no simple way to unstack the boards so you can reach in and check individual ICs or swap them.

There's much to be said for the plug-in card system, complete with an extender board for testing and service. They may be doing well enough with industrial orders, but if they do decide to go after the growing hobby market, a nice cabinet and plug-in cards would help a lot in competition with other systems now available.

IMSAI

Our next visit was to the new Imsai plant in San Leandro where we had a chance to talk with Ed Faber. Ed said they were getting ready to move to an even larger building and declined to let me see their present plant or take any pictures. Maybe next trip.

GODBOUT

Having been to visit quite a few surplus dealers, the mountains of stuff at Godbout's didn't surprise me. A closer look at some of the boxes made more of an impression . . . for one was a large box full of plastic tubes, each packed with 8080A chips . . . another box was full of 6800s . . . that shelf over there was piled with boxes of RAM memory chips . . . the next shelf with PROM chips, and so it went!



Here's Bill Godbout, just in case you have the idea all millionaire businessmen dress in pin stripes. How he provides the fantastic service he does from the mountains of parts is one of the wonders of the world – yet readers are constantly writing in complimenting Bill. Not a gripe in a carload.

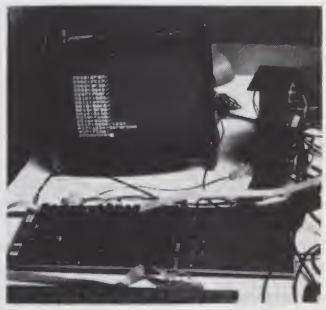


Godbout could probably use a computer just to keep track of the zillions of parts and chips he has around. One of the boxes on the left has about a thousand of the newest, fastest, hottest 8080As . . .

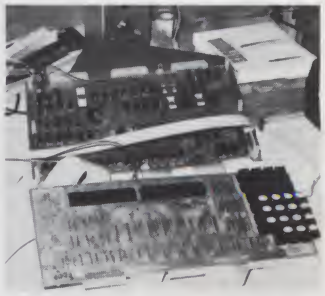
After going through several rooms full of equipment and parts, we came to one with a table and a test setup of Godbout's new PACE computer system. I gather that this system has been undergoing constant update modifications while Bill Godbout has been trying to get firm commitments on chip deliveries to support it. Once he has everything either on hand or definitely coming down the pipeline, he plans to start releasing the system . . . a very well designed and supported 16-bit microcomputer.

We all drove over to visit George Morrow and see how he was coming along with his stuff . . . a front panel for the Altair/Imsai systems . . . an operating system, monitor, debugger, and so forth for the Godbout PACE . . . and a lot more projects. George is one of those people with super energy – full of ideas – unable to sit still for a minute – talking enthusiastically about this and then that – more ideas – then off to work on six to twelve different projects simultaneously.

Bill and Reo (Reo Pratt, who works for Bill – pilots Bill's planes – writes articles for *Kilobaud*) managed to talk George into coming with us for dinner. While you and I might think it a big deal to drive 25 miles or so for a good dinner, our group went back to the airport at Oakland and flew about a half hour to get dinner. George talked animatedly during the flight up and back and all through dinner . . . we had to move the glassware back to keep him from sweeping it off the table as he waved his arms to illustrate points every now and then. Not that anyone discouraged him in any way, for his ideas were always interesting and provocative. I sure wish he'd write.



George demonstrated the debugger program he has in his front panels. It certainly is flexible.



This is the PACE kit which George has designed for Godbout. The lower board is the front panel board. George's front panel replacement for the Altair is meeting with success since it contains excellent soft and firmware.

Around the Industry



A few hours with George Morrow is an experience. This genius is coming up with great ideas several times an hour. It would take a big plant and thousands of people to keep up with his creativity. George does the design work for Godbout and for his own Morrow's Micro Stuff.

APPLE

A small ad had appeared for the Apple computer so we drove up to see what this was about. We met Steve Jobs and his partner Steve Wosniak, who have developed a first rate computer system in one corner of the Job's garage. Jobs did most of the hardware and Wosniak has done the operating system. Not bad for a pair of 20-year-olds. I was quite impressed by the thinking that went into the Apple ... the whole computer and video generator is on one board, with a very small cassette interface board which plugs into the main board. It is simple, yet elegant.

When you consider all of the engineering that goes into a computer, the fellows have done a fantastic job. Now they are getting into serious production — signing up dealers — and they will have their hands full.



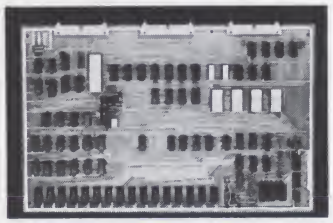
This is the Apple computer running a game of Life — the small board sticking up at the right is the cassette interface!

M&R ASTRAL 2000

Not far from the Apple "plant" we visited the M&R firm where they were putting the finishing touches on a most interesting system. It is particularly well designed mechanically, with cards which slide in solid guides plugging into a mother board which is just back of the front panel of the system. The front panel plugs into the mother board, making a very solid mechanical arrangement. This is one of the best mechanical systems I've seen yet.

The size of the individual cards is a bit smaller than I'm used to, but the chaps at M&R have those chips packed in solidly. The front panel mounts in a 19" wide cabinet, so there is no problem in expanding the system to accommodate a lot more cards with a second chassis which can mount right above the first. The power supply is very hefty and should handle a lot of cards.

They were particularly proud of their software — you'll be reading about it. It's an 8K Basic with a lot of interesting features. Since there aren't all that many Basics for the 6800, this will be most welcome. It comes with a Star Trek game and a newsletter for $35.



Here's the PACE CPU which should be available soon from Godbout.

WAVE MATE

Dennis Brown has a nice plant down in Los Angeles where he turns out the Wave Mate systems. These are the only wire-wrapped computers in the hobbyist market, so we asked Dennis a lot of questions about that. The answers were interesting, so we prevailed on him to write an article so you'd know the reasons for using this more difficult type of construction.

Wire-wrapping certainly is used to advantage in the Wave Mate ... all of the circuit boards are identical ... a design Dennis invented which permits almost any configuration you could want. One big plus with this system of building is that changes are a lot easier to make. Dennis has recently come up with a Z80 based CPU. With most other systems you'd have to try to sell or throw out your old CPU board and put in a whole new one to go with the Z80 — but with the Wave Mate this merely means some wire changes — not a big deal.

Dennis was working on an economy model of his system. The higher price of the Wave Mate has kept sales down, which is unfortunate, for in most of these things you get what you pay for. If you buy cheap you get cheap. For instance, as far as I know, Dennis is the only manufacturer who checks out each and every chip before sending it along for a kit or assembled unit. While not more than a small percentage of chips are faulty, a two percent fall-out will give you a good chance of at least one bad chip in a 50 chip system. It's difficult enough to troubleshoot a system that you've had working, but trying to find a bad chip in a new system is real bad news.

COMPUTER MART

We stopped off in Orange to see John French and his Computer Mart operation, probably the largest computer store in the world. John was the one who made the good deal for the Lear Siegler terminals which a lot of us are using. He was well along toward being able to market a 16-bit computer — watch for announcements on this one. We got promises of several very good articles from them. Dick Wilcox stopped by at the right time and promised articles on monitors, operating systems, etc.

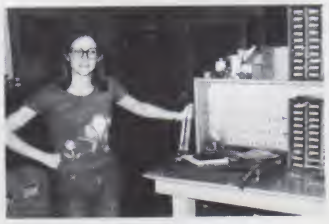
SPHERE

From Los Angeles we flew up to Salt Lake and an update on Sphere. Mike Wise, with whom I talked at length last year, had been replaced by a new president of the firm. I did get a chance to talk with Doug Hancey, who had been there the previous year. They've had a lot of problems getting delivery on parts and their plans for providing a good fast version of Basic had misfired. Doug was quite enthusiastic about their new TPU board — and they are making deliveries on their regular Sphere 6800 based systems.

Sherry was particularly happy to get to Salt Lake for it gave us a chance to see her sister and have dinner with Sherry's son, Mat, and daughter, Abra. We'd had dinner with her other son, Roger, in Los Angeles.



Dennis Brown, the designer and president of Wave Mate.



Allison Martin, Dennis' super assistant who does much of the production work.

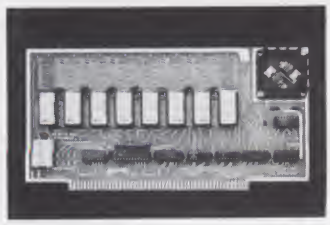


The Sphere kits are furnished with every part clearly labeled and stuck into a foam sheet. Missing parts are immediately apparent with this simple system — and it sure makes it easy to locate the right parts for the right holes on the boards.

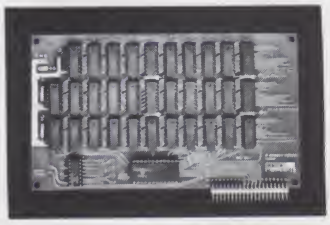
We managed a short get-together with the chaps who make those MPI printers and got a promise of an article from them. We have one of the printers in the *Kilobaud* lab and it does a grand job. We were a little short of time so didn't get out to their plant.

MITS!

It would take several pages just to outline the many projects going full tilt at MITS these days. Ed Roberts had cleared the day for us and we talked for hours on the hobby computer business, about the plans he had for further Altair products ... and about ham radio. Ed has gotten all fired up about getting active on amateur radio again and he had a lot of questions about that. This is not one of my weak spots.



How about this 8K EROM board? Designed by George, available from Godbout.



This is one of those Naked Ram boards — for a 6502 based CPU system.

Around the Industry







MITS has a whole big building full of people working hard to turn out the Altairs.

To give you an idea of how enthusiastic he is — not long after I got back home, he got a Kenwood 820 transceiver and a slow scan television monitor. Within a couple days he had an interface for the slow scan monitor so it would print out on a Teletype machine! Think of how much money hobbyists could make with an Altair system to print photographs from a slow scan camera system! It should cost about 10% of what the present computer photo systems now cost.

Ed did not share my reservations about the problems of software for the Z80 chip and explained that they were just about ready to announce a new Z80 CPU for the Altair. There's an article on this in this issue. He doesn't think that rewriting the software for the Z80 will take nearly as long as it did for the 6800.

MITS is well settled into their new plant now and things seem to be moving along rapidly, with several programmers developing their disc operating system, Basic for the Z80, and updates on everything else. They're also working on some new low cost floppy systems and have some other ideas in prototype shape in the back rooms. It's not likely that anyone else in the field is going to catch them sleeping.

SOUTHWEST TECHNICAL PRODUCTS

Last year when I visited Southwest, Gary Kay, the chief engineer, had a prototype of the SWT6800 computer system going, with hopes for delivery along in October or November. Now, several thousand kits later, with their system the largest selling of the 6800 based computers, they were putting the finishing touches on details of their graphics package and their newly-announced $250 line printer kit.

Dan Meyer, the president, took us all through their large plant — their main business has been high fidelity audio kits for some years, but now the computer orders have been gaining rapidly. We saw the new printer work, with its 40 character printout on a kind of wide adding machine tape. This is fine for dumping programs and other such things which you might want to save. It's a bit restricting for writing letters — but if you can make do with capital letters, a five by seven pin matrix print of the letters, purple ink and narrow paper, you're all set. The price is certainly right!

They had their new cassette system running well too and used it to load their graphics program. I'll surely have to get one each of those. After playing the tank games in the quarter arcades, I can't wait until I am able to get one of them working for myself; and this graphics system looks like the way to go.

We got a short demo of the 4K Basic they have at present, and they promised 8K would be available soon. They told the truth because I do at present have their 8K in hand.

That evening Sherry and I drove out to visit her folks in Seguin, just east of San Antonio. I've known them for a long time, but hadn't seen them in over 15 years; so it was a great reunion. I love to cook and one of my favorite dishes is a curry I learned from Alma, Sherry's mother. She served it with about ten different condiments ... like crushed pineapple, chopped nuts, coconut, raisins, chopped egg, etc. Fantastic.

MOTOROLA

Since we weren't far away, we stopped off for a few hours in Austin to visit the new Motorola LSI plant. We were given the red carpet treatment and got to see them making LSI chips. I'm not sure an article would really get the idea across of the magnitude of the equipment and the care needed to turn out these ICs. They grow the silicon crystals, then turn them on a lathe, slice them into thin slices, polish the slices, and then, step by step, deposit the layers of the transistors and circuits on these slices of silicon, each about 3" or so in diameter. When they get through they cut the tiny IC chips up and check them.

There are a lot of interesting ICs in the works — like 32K RAM chips and perhaps even a 64K RAM! One recently announced product is an inexpensive digital voltmeter chip (DVM). This is really just an analog to digital converter, but it comes complete with the coding needed to produce numbers on LED readouts.

THE USS DRUM

Since my old World War II submarine crew was holding a 30th reunion at Mobile where our old boat is on display in the Naval Museum there, tied up alongside the Alabama, we stopped off there for a day. All those youngsters who fought the war with me in the 1940's are all old men! What a bummer. We had a great time talking over the War and we have reason to be proud as we were on one of the top submarines in scoring. But we didn't talk much of ships sunk or prisoners taken, but rather our memories of a cook whose spaghetti fell off the stove and who was caught trying to put it back in the pot — the humorous anecdotes of the War.

INTELLIGENT SYSTEMS

Our last stop on the trip around the country was Atlanta to see the outfit making that epitome of color graphic terminals, the Intelligent Systems CRT terminal.

We found the plant just on the outskirts of Atlanta and had a nice tour. I expected a bit more business than I saw — perhaps we can help them get the word around. Their ads give the impression that you can have a full color terminal for probably not a lot more — maybe even less — than a regular black and white terminal.

The Intel color terminal is certainly reasonable, but as with any other computer system, you have to decide which accessories you really can't do without — and so many of the alternatives available with this unit seem desirable that you can run the bill up pretty steep. The system comes with an 8080 processor built in — and this means that you have a lot more than a dumb terminal, you have a full computer, if you want to go that route.



Terry Huey, president of Intelligent Systems.

You can call up the letters in any color you want ... and with one of their confounded accessories you can have a background in one color and the letters in another ... it's gorgeous. Sure, it's a little larger than the black and white CRTs, but why not go full color?

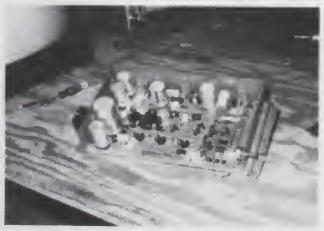
While the unit is pretty expensive by hobby standards, particularly if it us used just as a terminal, it may go well in commercial use once the industry gets familiar with it. We'll know more about that when we have some programs available to run with it — it could turn out to be a great home terminal unit and computer.

We made a stop in the Computer Systemscenter in Atlanta and had an interesting talk with the four chaps who are running it. They are selling

continued on page 24



Some of the Intel Systems being run for test — they are checked for a week before being shipped.



There's a lot in the box — this is the analog module, for example.



The computer Systemscenter has some very good software available for business uses.



Selling to business? Then you need a nice looking work station like this, complete with terminal, floppies, and printer. The Systemscenter has several good business programs available and they are working on a lot more. This may be a very good model of what a computer store will look like by next year.

BOOKS

from page 11

wheel control problem, the reader could then benefit from the chance to try these skills on some additional examples, even if treated briefly. Some cases involving more data manipulation and/or more complex input/output procedures would provide a more complete picture of the 8080 in the systems environment. However, the author may have felt, with some justification, that this went beyond his purpose of considering the 8080 in logic design. This is a minor and personal objection and should not discourage anyone interested in Mr. Osborne's excellent book.

A. H. McDonough
El Segundo CA 90245

The Best of Creative Computing, Volume 1, David H. Ahl (ed.); Creative Computing Press, Morristown, New Jersey, $8.95, paperback, 8½x11, ix+317 pages.

If you are a junior high or high school teacher with any inclination at all to teach about technology (doing it, using it, analyzing it, or living with it), and you haven't seen *The Best of Creative Computing, please* drop whatever you're doing and go get this book. If you know of such a teacher, get him or her a copy, and make sure that it is read.

But how about the rest of us? What does a volume, edited by a man whose long-term mission seems to be to inform, reform, and revitalize secondary school teachers, their methods and curricula, offer to us? That is, what does this collection offer to people who want or have their own computers — who are interested in putting them to use in their daily lives, not in the classroom?

Quite a bit, it turns out, but much of it is at a fairly high level (meaning remote from the hardware, remote from specific programming techniques). From the aspects of being informational, well-stocked with ideas to draw from, and entertaining, this volume is virtually unbeatable. Certainly nothing in *my* local computer store has anything that can compare with *The Best of Creative Computing* in the softer, more human (as opposed to hardware) reaches of the computer world.

There are seven main sections — let me cover them one by one. Since something about the book made me want to read it from the back, I'll go through it in that order.

On the back cover is the great R. Crumb "Don't you think it's time to Stop Watching TV?" ad, remade (by the artist) into the "*Creative Computing*" ad you've no doubt also seen. (It originally appeared on the back of *Motor City Comics, No. 2.*)

The last section in the volume contains reviews of over 70 books, many of which are of interest. You may have to read between the lines to find what you're looking for, but there is a lot of useful information here.

The next to last section has the most to offer if your goal is discovering things to run on your machine. It's called "Computer Games," and it includes program listings (in various dialects of BASIC) for 17 games including two of the all time favorites, Hunt the Wumpus and Star Trek. One of the games (Geowar) is followed by a critique by Gregory Yob (creator of Hunt the Wumpus). It's must reading for prospective game designers. In fact, the whole notion of reviewing and critiquing games and software in general is important. We should see more of it.

The fifth section is "Programs, Puzzles, Problems, and Activities." Although most of the articles are aimed at students in a classroom setting, there are a lot of interesting ideas, projects, and an occasional program. There are a number of gentle mathematical topics, programs which help young users write poems and stories, several suggestions for attitude surveys, articles on simulations, a program which can be used to study particles falling under gravity, and a program which can be used for word drill in a foreign language. All in all, a wide variety of fascinating topics, well presented. I wish Creative Computing had been around when I was in school!

Before that you'll find "People, Places and Things," which has a Whole Earth flavor to it. It includes an access to media section (the Compleat Computer Catalogue) which lists a number of relevant books, magazines, newsletters, etc. Well worth a look. You're almost certain to find an intriguing title you never heard of before. There's also a piece of pragmatic knowledge in this section — tips for winning at Pong.

"Foolishness" contains several humorous short articles and six full pages of comics. All are funny, although they all seem to have been done by people from the outside looking in on the computer world.

The section called "Fiction and Poetry" contains three short stories and a page of poetry. The Isaac Asimov story "The life and times of Multivac" makes pleasant reading, although oddly enough it seems a little dated despite its 1975 copyright.

The first and longest section in the book is "Articles and Commentary." It covers a great variety of topics, many of great timeliness and many relevant to anyone who hopes to cope with our computerized society. Again, although much of the material is aimed at educators, there are lessons for all of us. For instance, the two articles on program portability ("The Parable of the Horse" and "Technical Transport Problems") should be seen by anyone who is writing software they hope to share with others. The specific problems discussed may not be directly relevant to the home computer situation, but the general difficulties and the solutions definitely are.

Two Overall Impressions

One thing that really grabbed me when I looked through this volume is the artwork. Virtually every page has some kind of nonverbal visual pattern on it, most of them very neatly tied to the ideas in the text matter. A fantastic job! And the artists! Not just R. Crumb, but other masters too — an absolutely incredible drawing by Bill Elder (p. 117), and several things by Will Eisner. The drawings, computer graphics, and photographs just don't stop! Flipping through, I see page after page with well thought out graphs, cartoons, several fine pieces by George Beker, several derivative but well-executed drawings by the editor himself, a Gilbert Shelton, a Shary Flenniken, and, on the front cover, a computer printout picture of our favorite Vulcan.

I *was* left with one sort of puzzling feeling. David Ahl obviously has a great concern and care about educational reform and for loosening some of the restrictive, stultifying aspects of the education establishment (EdBus) . . . but there seems to be little consideration of anything *but* formal education here. Isn't it conceivable that the Sesame Street idea coupled with the home computer may move the task of preparing our children to live in our society out of the formal classroom and back into the home? Just a thought.

In Summary

Does *The Best of Creative Computing, Volume I* have anything to offer the computer hobbyist? If you're looking for hardware hints, no. If you're looking for software tips, maybe. If you're looking for games, yes. And if you're in search of the Big Picture, absolutely.

PACE: Logic Designers' Guide to Programmed Equivalents to TTL Functions; National Semiconductor Corporation, 2900 Semiconductor Drive, Santa Clara, California, 95051; Order Number IPC-16A/927; Publication Number 4200127X; $5.00

Some years ago, when the minicomputer first appeared, the conventional wisdom stated that these devices would generally replace special purpose logic assemblies and that logic designers would metamorphose into programmers. However, considerations of speed and cost intruded and the impact of the minicomputer on special purpose one-of-a-kind logic was never as great as expected. Now the same sort of impact on special purpose logic design is being attributed to the microprocessor, a much less costly device.

This book is National Semiconductor's attempt to promote the microprocessor, in particular their PACE unit, as the logical successor to special logic arrays and to aid the logic designer in the necessary transition to part-time programmer. The title of the book is descriptive of both the contents and the intended readership.

The book begins with a brief introduction to the basic concepts of the digital computer in general and of the microprocessor in particular and to the general concepts of software. The hardware-oriented sections are very sketchy and obviously assume that the reader is fairly knowledgeable and really needs little introduction to the subject. The software section, however, assumes that the reader has little or no familiarity with the various levels of programming and does an excellent job of introducing the concepts of assemblers, translators, loaders, and the programming process. This is all done in an informal tutorial style that is easy to read and to understand.

The introductory material is followed by two sections on the particulars of the PACE unit. One lengthy section describes the instruction set and a shorter section provides information on input/output. The instruction set material is typical in content and format of any computer reference manual. This can be scanned once and then used as reference data when required. The input/output section is brief but covers the material adequately and does a good job of describing the interrupt system if you have some hardware background.

Special purpose logic of the type treated in this book rarely exists by itself. It may be driving the light bulbs of a simple visual indicator or controlling a complex piece of scientific apparatus but special logic is inevitably interfaced to something. This book is therefore of little interest to the user whose microprocessor is dedicated to pure computation or to computer games and which ties only to one or two standard peripherals.

The book is aimed at the working hardware designer with little or no software background but with a sound basis in logic, a fact explicitly stated in the title. For that somewhat narrow readership this is an excellent book. For those without the grounding in logic devices it is likely to be rather difficult reading. ■

A. H. McDonough

Microcomputer Design **by Donald P. Martin; edited by Kerry S. Berland; Martin Research, 3336 Commercial Avenue, Northbrook IL 60062; 390 pages; $25.00.**

This is an advanced book with a wealth of detail for the hardware designer who intends to start with 8008 or 8080 chips and fabricate a complete microcomputer. It should also be of interest to the owner/user of a microcomputer with the ambition and the technical ability to undertake major additions or modifications to the system. However, the technical content and the very detailed design data do concentrate far too heavily on the 8008. While the 8008 is not actually obsolete it is no longer state of the art and is unlikely to be the device selected for new applications.

The book includes as an appendix a set of 8008 and 8080 data sheets. In the preface the author suggests that

continued on page 24

Pat Godding
Vice President
MITS, Inc.
2450 Alamo S.E.
Albuquerque NM 87106

Is the Z80 the Wave of the Present?

Step right up, folks! See the mighty Z80 do battle with the 8080 and the 6800! Pat Godding has presented us with some interesting facts and some comparisons of these three popular microprocessors. Now, I'd like to see someone run some tests with the three (both hardware and software evaluations) using game or small business routines (or whatever) and see which is really fastest, most efficient, most easily programmed, etc. — John.

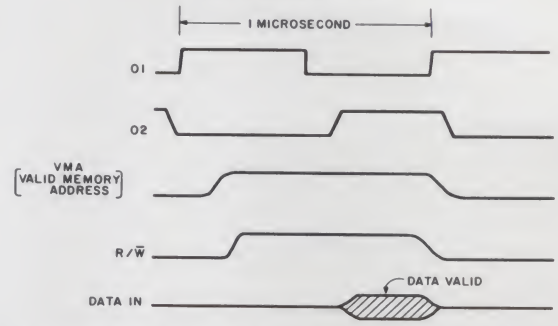
The general purpose microcomputer has found an extremely large market (both industrial and hobbyist). This is not surprising since there are a vast number of applications where a minicomputer is both an overkill and much too expensive.

It is the size of this market that caused the other main manufacturers of integrated circuits to follow Intel's lead. Since the Intel 8080 processor chip was the first to be used in a general-purpose computer system, some of the manufacturers chose to produce pin-for-pin replicas. A different approach was taken by Motorola. This was to simplify the external hardware required to support the processor. This latter approach became the 6800.

Although the 6800 is easier to interface and has a more flexible instruction set, it's clock speed is about one half of the 8080's clock-speed, and it contains no separate input/output structure (I/O devices must use memory locations for communication).

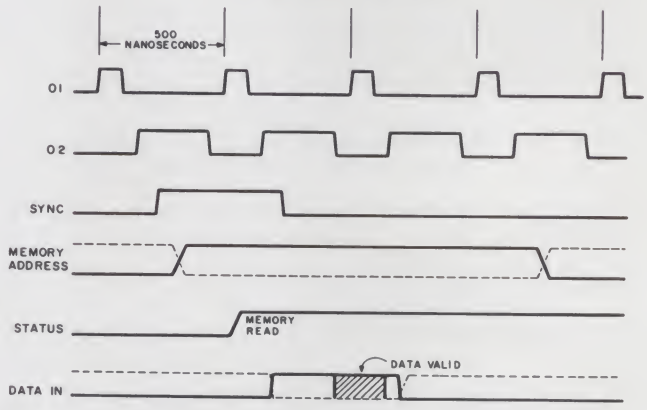
Although many new processors have become available, some faster, some less expensive, and some requiring less interface hardware, none possessed any earth-shattering breakthroughs — until the Zilog Z80 was announced several months ago.

Motorola 6800
(clock cycle = 1 usec)



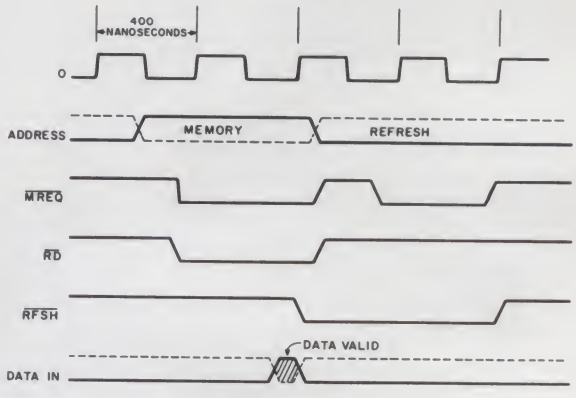
Note: A *minimum* of 2 of the above cycles are required to execute a single byte instruction (2 microseconds).

INTEL 8080
(clock cycle = 500 nsec)



Note: A *minimum* of 4 cycles (each 500 usec) are required to execute a single byte instruction (2 microseconds).

Zilog Z80
(clock cycle = 400 nsec)



Note: A *minimum* of 4 cycles are required to execute a single byte instruction (1.6 microseconds).

Fig. 1. Timing diagrams for 1 machine cycle (READ).



Fig. 2. Status information latching circuit.

The Z80 incorporates the best features of both the 8080 and the 6800, and adds some very powerful features of Zilog's own creation.

External Hardware

The 8080 requires two clock phases and three voltage supplies (+5, -5, and +12 V). The 6800 simplified this by going to a single +5 V supply but at the expense of reducing the maximum clock frequency by one half. The basic 8080 clock frequency maximum is two megahertz; for the 6800 it's one megahertz.

The Z80 uses one clock phase and a single +5 V supply, and the maximum clock frequency for the basic Z80 is 2.5 MHz*! This is a phenomenal speed for MOS LSI technology (Metal Oxide Substrate, Large Scale Integration) since the speed of a MOS LSI device generally suffers when the voltage is reduced, as seen above in the 6800 processor.

Fig. 1 shows timing diagrams for the 8080, 6800 and Z80. The 8080 "talks" to the external world by multiplexing its data lines. The term multiplex in this description simply means that the data lines are used for more than one purpose. The different purposes never conflict because they occur at different times during any given cycle. At the beginning of each cycle, the 8080 sends out a signal called SYNC. When this signal is coincidental with the phase one clock (01), the data lines contain status information. These status signals tell the external hardware what type of cycle is about to be executed, e.g.: read from memory (MEMR), read I/O (INP), write to I/O (OUT), etc. These signals are used by the memory blocks and device interfaces so that only one memory location or I/O device is selected during any given cycle. To maintain the status information throughout the cycle it must be latched (stored) as shown in Fig. 2. Thus, each time a SYNC pulse is generated, the latch will be updated to contain the new status information. The primary advantages of this multiplex scheme is that it effectively increases the number of signal lines available on the 40-pin IC package. The number of pins available for control signals is a problem when there are already 29 pins used for address, data, supply, and clock lines. By effectively increasing the number of signals available, the 8080 is able to use an I/O structure separate from memory. The 8080 can access 64K words of memory (1K = 1,024) and 256 I/O devices. The 6800 is limited to accessing memory only (also 64K words) and uses memory locations for I/O device interfacing.

On the other hand, the 6800 is simpler to interface since it does not multiplex any lines; if a read cycle is being executed, the R/W (read/write) signal is in its high state coincidentally with the correct address.

The Z80 uses the same simple type of control as the 6800 but maintains the 8080 type of I/O structure. By gaining the four pins used on the 8080 for SYNC, one clock pulse and two voltage supplies, the Z80 eliminates the need for multiplexing. This makes hardware implementation easier and, even more important, it allows memory to be accessed earlier in a cycle, so that slower memory chips can be used (slower memory is less expensive). Two signals, MREQ (memory request) and IORQ (I/O request) are used to select memory or I/O devices, respectively. Two other signals, RD (read) and WR (write) define whether the memory or I/O will be read or written, respectively. This structure simplifies system design.

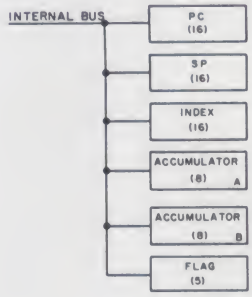
Zilog has also included a very important feature not found on other microprocessors. Dynamic Random Access Memories are relatively inexpensive, low in power, and fast, but they require constant clocking, called refresh, to maintain the internal charges that represent data. This must be accomplished without the processor "knowing about it." The ideal time for refresh to occur is when the processor is working internally. Each time a new instruction is fetched from memory, the instruction is decoded in the processor in the last part of this fetch cycle. This period is called Machine Cycle 1 (or M1), clock cycle 4 (or T4). Thus, a controller for dynamic memory in an 8080 system must decode this M1, T4 period to provide a refresh signal. The controller must also keep track of the refresh address and present it coincidentally with the refresh signal.

The Z80 has its own internal dynamic memory controller. During each M1, T4 period, the Z80 provides a refresh signal (called RFSH) and also places the next sequential refresh address on the lower 7 address lines (4K dynamic RAMS require the lower 6 address lines for refresh and 16K RAMS require the lower 7 address lines). Thus, the memory boards can contain less control logic and more memory.

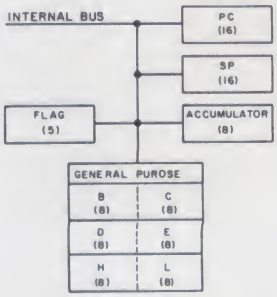
Internal Hardware

The 8080 contains 6 user accessible registers which are each 16 bits wide. (Refer to Fig. 3.) Three of these registers are general purpose and can also be used as six individual registers that are 8 bits wide. The remaining three registers are the Program Counter (PC), the Stack Pointer (SP), and the Accumulator/Flag. Most arithme-

MOTOROLA 6800



INTEL 8080



ZILOG Z80

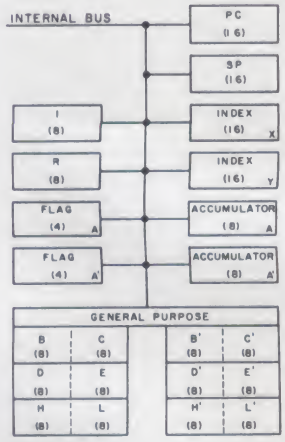


Fig. 3. Internal block diagrams illustrating user accessible registers. (Number in parentheses indicates the number of bits in a register.)

*Note! All three processors are available in faster versions.

tic and logic functions use the 8-bit Accumulator and the status of the results are stored in the 5-bit Flag register. The 6800 is similar, except that it contains an index register for added software flexibility and has no general-purpose registers. Also, the 6800 has 2 accumulators.

To maintain software compatibility, the Z80 includes the 8080 register bank as a subset. In addition, the Z80 has seven more 16-bit registers! Three of these are general purpose and identical to the other general purpose registers. A single instruction allows the programmer to access either register bank. The remaining four registers are a second Accumulator/Flag register, two index registers (IX and IY), and an Interrupt/Refresh (IR) register.

The 16-bit index registers normally contain base addresses used for building tables of data. The 8-bit Interrupt register holds the upper address for mode 3 interrupts (see Interrupt). The 8-bit Refresh register is the address counter used for dynamic memory refresh. Needless to say, with these additional accessible registers and an increase in the internal "housekeeping" logic, the Z80 has a much greater instruction set.

Instruction Sets

The machine language instruction set is physical hardware. A program, or group of machine instructions, is software. When a machine instruction is executed, the states (logic zeros or logic ones) of the 8 bits contained within the instruction are decoded by internal logic to define what function the processor must perform. Generally, when more instructions are included in the machine instruction set, it takes fewer of these instructions to perform a given task. This decreases the amount of memory needed for a given program and the program will normally execute faster.

The 8080 has 78 machine instruction *types*. An example of a machine instruction type is REGISTER INCREMENT. An operation code (op code) is the actual bit pattern that represents a machine instruction and each one is a unique group of 8 bits. Since the 8080 has six 8-bit registers and an accumulator, all of which can be incremented, there are 7 distinct op codes (or bit patterns) representing this one instruction type. The 78 instruction types of the 8080 constitute 244 op codes. The 6800 has 72 instruction types with 197 op codes.

The Z80 contains 158 instruction types, including all of the 78 instructions of the 8080. The total number of op codes is 696! Two of these additional instructions greatly increase the software speed of the processor. The first type is the block instruction. This allows blocks of data contained in one section of memory to be moved to any other section by executing a single instruction. The block size can be anywhere from 0 to 64K. Blocks of data up to 256 words can be transferred between memory and I/O devices. In addition, a block of memory can be searched for a particular bit pattern defined by the accumulator. The search continues until a match is found or the specified search length has been reached.

The second instruction type is bit manipulation. This set of instructions can set, reset, or test any bit in a register or memory location. The normal instructions required by the 8080 are used to load the accumulator and perform a logical AND with the specified bit. A large program may require this type of function many times.

Addressing Modes

Data is stored in internal registers, memory locations, and/or I/O devices. The machine instruction must obtain the address of the data in order to access it. Each address mode obtains the address in a different manner.*

The 8080 has 4 Addressing Modes:

a) Extended

The op code is followed by 2 bytes that specify the address of the data. (Exp: Jump to the address specified by bytes 1 and 2.)

b) Register

The op code specifies a register or register pair that contains the data. (Exp: Load the contents of the B register into the Accumulator.)

c) Register Indirect

The op code specifies a register pair that contains the memory address of the data. (Exp: Load the Accumulator with the contents of the memory location addressed by registers B and C.)

d) Immediate

The op code is followed by 1 or 2 bytes of data. (Exp: Load the Accumulator with the data *immediately* following the op code.)

The 6800 has 7 Addressing Modes:

a) Extended

b) Immediate

c) Accumulator

Specifies 1 of the 2 accumulators (e.g., increment the B accumulator.)

d) Direct

Same as extended, except only 1 byte specifies the address (limited to addressing only the lowest 256 words of memory).

e) Indexed

The op code is followed by 1 byte that is added to the index register. This sum becomes the address. A temporary register is used for the addition so that the index register is unchanged. (Exp: Load the Accumulator with the contents of the memory location specified by a number (0-255) plus the index register.

f) Implied

The op code implies the location. (Exp: Add the B register and the Accumulator. It is implied that the result will be stored back into the Accumulator.)

g) Relative

The byte following the op code is added to the program counter (PC). This new address is then jumped to if tested conditions are met. (Exp: If Carry is set, jump to PC + 64.)

The Z80 has 8 Addressing Modes:

a) Extended

b) Register

c) Register Indirect

d) Immediate

(*Note:* The above 4 modes maintain software compatibility with the 8080.)

e) Indexed

f) Implied

g) Relative

h) Bit

There is a large group of instructions that can set, reset or test an individual bit within a memory location or register. (Exp: Reset bit 2 in the memory location specified by registers H and L.)

In addition to the above modes, instructions may use two or more combinations of these modes for execution.

Interrupt Structure

Multilevel interrupts in the 8080 are provided by the RST (Restart) instruction. Three bits of this instruction define which of eight possible locations will be jumped to when the processor receives an interrupt signal. These three bits are provided by an interrupt control module which sets priorities for up to eight incoming I/O interrupt lines and allows only the highest priority to interrupt the 8080.

The 6800 has two modes of interrupt. The first mode requires that the interrupt enable instruction has been executed. Then an incoming interrupt causes a jump to a location defined by the contents of two bytes of memory (FFF8 and FFF9). The other mode is similar except that it

*Note: The 8080 and 6800 use 1 byte (8 bits) to contain each op code. The op code may require additional bytes for execution. These are either 1 or 2 bytes immediately following the op code. The Z80 uses the same instruction format, but either 1 or 2 bytes may be used to contain the op code.

TELETYPE MODEL 33



TWX AS SHOWN
$1,450.00

TWX or COMPUTER INTERFACE

$840.00

- 33ASR PRIVATE-LINE
- FRICTION FEED
- COPYHOLDER & STAND
- ANSWERBACK
- MANUAL READER
- GUARANTEED 30 DAYS
- F.O.B. NEW JERSEY
- CRATING INCLUDED
- NOTHING ELSE TO BUY

Options:

- AUTOMATIC READER ADD $50
- READER RUN CARD (DEC) ADD $75
- SPROCKET (PIN) FEED ADD $100
- TAPE WINDER (ELECT.) $55 - WINDUP $22
- EIA INTERFACE $110
- TAPE UNWINDER (NON-ELECT.) $33
- PAPER WINDER (ELECTRIC) $50

master charge

Your BANKAMERICARD welcome here

--- NEW FREE CATALOG AVAILABLE NOW ---

TELETYPEWRITER COMMUNICATIONS SPECIALISTS, INC.

550 SPRINGFIELD AVENUE
BERKELEY HEIGHTS, N. J. 07922

PHONE - 201-464-5310 TWX - 710-986-3016 TELEX - 13-6479

BUY * SELL
SERVICE * LEASE

* OVERHAULING & MODIFICATIONS
* REPLACEMENT PARTS
* PAPER--TAPE--RIBBONS
* VIDEO TERMINALS
* DECWRITERS
* ACOUSTIC COUPLERS

uses address locations FFFC and FFFD and does not require the enable interrupt instruction to be executed.

The Z80 has three modes of interrupt. The first mode is identical to the 8080. The second mode immediately jumps to location 38 when an interrupt is received (and the interrupt enable instruction has been executed). The third mode is by far the most flexible. When an interrupt is received, the processor forms a 16-bit address, in which the contents of the interrupt register constitute the upper 8 bits [this register is loaded via software] and the I/O device supplies the lower 8 bits. This 16-bit address represents the memory location that contains the actual address of the interrupt service routine for the device. Thus, the device service routines can be located separately from each other anywhere in memory, and the addresses for these routines can all be contained together in a table whose starting address can be at any location in memory. This indirect addressing scheme is called vectoring.

Conclusion

The Zilog Z80 is much more powerful than either the 8080 or the 6800. There are certainly a large number of applications that need the power of a Z80. However, in a general-purpose, stand alone microcomputer system, the 8080 and 6800 type processors have proven to be very efficient. Perhaps the brief comparisons shown here will impress some personal computer users, but whether those users will be motivated to upgrade their systems, remains to be seen. And with the 8080 bus compatible Z80 CPU boards available (including one from MITS), we should see very soon. ■

BOOKS

from page 19

the "novice" begin by reading these data sheets and then reading chapters 2 and 3 which describe, respectively, the 8008 CPU and the 8080 CPU. An even better suggestion for the novice is to read some other text and gain some hands-on experience before approaching this book. *Microcomputer Design* is surely no book for the beginner. The twenty-five dollar cost alone would ensure that.

To add to the novice's problem the chapters on the 8008 and 8080 CPUs are by no means the best chapters in the book. The novice will spend more time referring to the data sheets and to other chapters, digging out data, definitions, and concepts, than he will reading the chapter at hand. The book is simply not organized as a tutorial text and attempting to use it as one will most likely persuade the novice to take up some other line of work.

However, starting with chapter 5 on Main Timing, the book is an excellent aid to the experienced designer who is now working with 8008/8080 microprocessor or microcomputer systems. The book covers a variety of topics (26 chapters and 390 pages) including bus structures, input/output, ROMs, RAMs, DMA and interrupts. Several chapters deal with the connection to the outside world, to peripherals, to analog devices and to special circuits which extend and expand the functions of the 8008. The last two chapters give sample designs for microcomputers, ranging from the smallest system worthy of the name to a relatively sophisticated unit using 19 chips.

Throughout chapters 5 through 26 the emphasis is on the specific and the practical. The circuits used to implement the various functions are clearly diagrammed with the appropriate IC chip number, and the operation and timing are discussed in detail. Some consideration is also given to the much neglected areas of cost and ease of manufacture, not a lot of consideration but at least some.

One small complaint is the relatively little attention devoted to software. One short chapter presents a few useful concepts, all of which are related somehow to hardware problems. It may be argued that this is, after all, a hardware design book. However, that simply perpetuates the isolation of hardware design from consideration of the end use. There is already too much of that.

That philosophical objection aside, *Microcomputer Design* should be a valuable book for an experienced designer working with 8008 and 8080 based systems. The emphasis is too much on the 8008 rather than the 8080 and hopefully the author will reverse that emphasis in future editions. Despite this, the book does cover a broad range of topics in admirable detail. For the beginner, however, a simpler book would be best. ■

A. H. McDonough

Around the Industry

from page 18

hardware — but even more important they have some good business programs that they have developed and are marketing. It's good to see computer stores working to sell microcomputers for business uses.

BACK HOME

It was the encouragement given by all of the manufacturers and dealers we visited during this 10,000 mile pilgrimage that resulted in *Kilobaud* getting started. We discovered that the magazines in the hobby computing field were not doing what a lot of the industry felt should be done — and there seemed to be agreement that my concept of a magazine which beginners could read was badly needed.

Yes, I realize that getting experts in this field to write so all of us can understand what they are trying to explain is going to be very difficult — and I know we're just not going to make it all the time, but we'll be in there pitching.

The first announcement of the new magazine to the public at large was at our booth at the PC-76 in Atlantic City. Apparently a lot of people thought my idea was a good one for we were mobbed and sold a lot of subscriptions.

My visit to the hobby computer industry was encouraging — some firms are doing very well, others are hurting. In general most of those who are not being rushed for deliveries have a credibility gap which has been generated either by vague advertising, poor deliveries . . . or both. There is no question that the field is not experiencing the growth which was expected, but this is not all that surprising for little has been done to bring news of the new hobby to the general public. Outside of the large number of articles in *73 Magazine* and a little bit in *Radio Electronics* and in *Popular Electronics*, there has been very little PR. The 75 or so recent articles in *73* on hobby computers has brought in tens of thousands of hams, but other than that there seems to have been little growth. This is something to think about. ■



The window "grabber" is an Altair controlled model train system. It sure brings them in.



The Systemscenter has the Intelligent Systems unit working, connected to an Altair 680 and a National Multiplex cassette system. Just about all of the programs available for this unit have been developed by the fellows here.



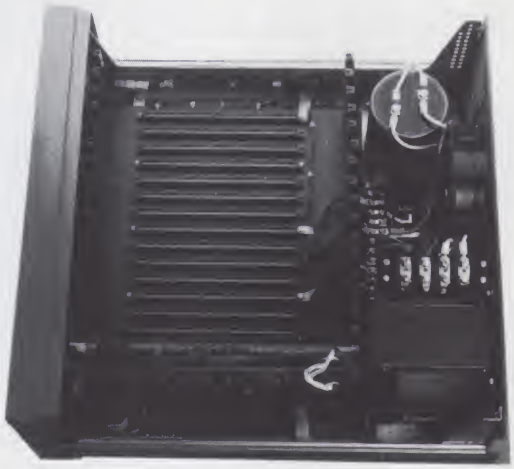
The Systemscenter is aimed at the hobbyist, the professional, and the homeowner. A Cromemco Dazzler with Life playing on it is on display (in color) near the window and it, too, keeps people watching — particularly at night.

mits catalog insert

2450 Alamo S.E. / Albuquerque, New Mexico 87106

altair™ 8800b

The ALTAIR 8800b computer is a general purpose byte-oriented machine (8-bit word). It uses a common 100-pin bus structure that allows for expansion with either standard or custom plug-in modules. It supports up to 64K of directly addressable memory and can address 256 separate input and output devices. The ALTAIR 8800b computer has 78 basic machine language instructions and is comprised of a power supply board, a D/C interface board, a central processing unit (CPU) board, and a display/control board.



NEW DESIGN FEATURES

Several new design features have been incorporated into the electronic and mechanical areas of the ALTAIR 8800b computer. Some of the new design features include additional front panel capabilities, redesigned power supply, and various electronic and mechanical design advancements.

New Front Panel Switches

Five new front panel switch positions have been added to the ALTAIR 8800b computer to expand the front panel capability.

1. SLOW : Permits execution of a program at a rate of approximately 2 machine cycles per second or slower.
2. DISPLAY ACCUMULATOR : Displays the contents of the CPU accumulator register on the ALTAIR 8800b front panel.
3. LOAD ACCUMULATOR : Loads the information present on the lower eight front panel address switches into the CPU accumulator register.
4. INPUT ACCUMULATOR : Inputs the information present at an Input/Output device into the CPU accumulator register. The Input/Output device is selected on the upper eight front panel address switches.
5. OUTPUT ACCUMULATOR : Outputs the contents of the CPU accumulator register to a selected input/output device. The input/output device is selected on the upper eight front panel address switches.

New Power Supply

The new power supply in the ALTAIR 8800b contains an 8 volt, 18 ampere tapped secondary supply which permits the addition of up to 16 printed circuit cards, and pre-regulated positive and negative 18 volt, 2 ampere supplies. A multiple tapped primary transformer provides for 110/220 volt operation and a 50/60 Hz operation.

Electronic Design Advancements

The electronic design advancements on the ALTAIR 8800b are in the CPU and front panel circuit boards.

1. CPU. The new CPU circuit board uses the Intel 8224 clock generator integrated circuit (IC). The 8224 IC provides a specified clock frequency to the ALTAIR 8800b using an external crystal and dividing the crystal frequency down to 2MHz. Therefore, both the clock pulse widths and phasing (as well as frequency) are crystal controlled.
2. Front Panel. All front panel data lines are connected to the D/C Interface Board which buffers them from the rest of the ALTAIR 8800b. The front panel circuits also use a programmable read only memory (PROM) which contains programs for the following eight functions:
 EXAMINE/EXAMINE NEXT
 ACCUMULATOR DISPLAY/ACCUMULATOR LOAD
 DEPOSIT/DEPOSIT NEXT
 INPUT ACCUMULATOR/OUTPUT ACCUMULATOR
3. The front panel circuits also have a wiring option which allows the CPU to perform a complete instruction cycle or a single machine cycle during the single step or slow operation.

Mechanical Design Advancements

The mechanical design advancements on the ALTAIR 8800b are incorporated for ease of assembly and maintenance.

1. Two ribbon cable assemblies connect the D/C Interface board to the Display/Control Board. Thus, the need for complicated bus wiring has been eliminated.
2. A single piece 18-slot motherboard has replaced the four slot expander cards in the ALTAIR 8800. The 18-slot motherboard contains 100 solder lands which comprise the 100 pin bus.
3. A new multi-color and redesigned dress panel is used in the ALTAIR 8800b. The front surface of the dress panel has a protective sheet of mylar to insure that the graphics are not rubbed or scratched off.

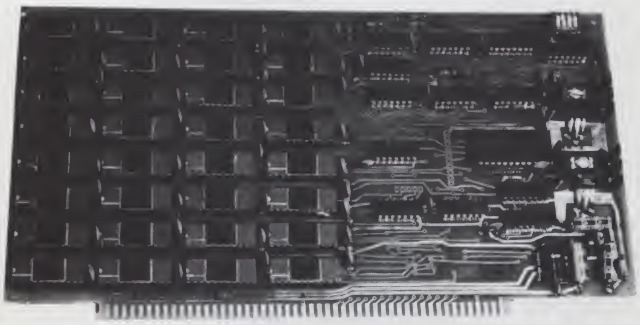
Compatibility

All of the current 8800 software and all of the current 8800 plug-in circuit boards are compatible with the Altair 8800b (with the exception of the 8800a CPU board).

ALTAIR 8800b
Specifications

Number of Boards	Up to 18
Microprocessor	
Model	8080A
Technology	NMOS
Data Word Size, Bits	8
Instruction Word Size, Bits	8
Clock Frequency	2M Hz
Add Time, Register to Register, Microsec. Per Data Word	2
Number of Instructions	78
Input/Output Control	
I/O Word Size, Bits	8
Number of I/O Channels	256
Direct Memory Access	Optional
Interrupt Capability	Std.
Vectored Interrupt (8 priority levels)	Optional
Software	
Resident Assembler	Yes
Higher-level Language	BASIC
Monitor or Executive	Sys. Mon.; text edit.
Complete Software Library Separately Priced	Yes
Price	
$840.00 kit	**w/2 edge connectors**
$1100.00 assembled	**w/6 edge connectors**

16K Static Memory Board (88-16MCS)



The 88-16MCS design is based upon a 4,096 x 1 bit static random access integrated circuit. These RAMs offer the advantage of low power consumption and extremely fast access time, and also eliminate the need for additional refresh logic common to dynamic memories.

Hardware

Dimensions: 10" x 5"

Altair Slots: One

Power: +5V (400 mA, Max.), +12V (200 mA, max.), −5V (100 mA).

Epoxy solder masks cover areas not to be soldered.

Sockets are provided for all RAM ICs.

A DIP Switch is used for board address selection.

Capabilities

Provides 16,384 8-bit words of Random Access Memory on one card.

Access Time: 215 ns.

Cycle Time: 390 ns.

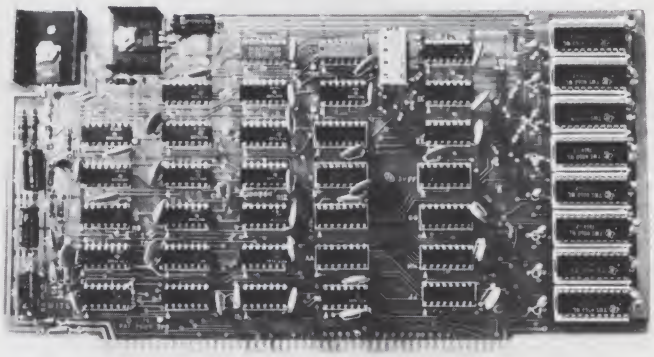
Operation

There are no wait states (due to fast access time).

Since only four 88-16MCS boards are required for the maximum amount of memory directly addressable by the computer, only the upper two address bits are required to select a particular board. The upper four address bits select a single 4K block on a particular 16K card. The remaining twelve address bits are buffered and inverted to select one location within a block.

Price: $765 kit
$945 assembled

Synchronous 4K Memory Board (88-S4K)



The 88-S4K Board is totally synchronous—this means that the board relies solely on the CPU for timing signals (no single shots). The S4K provides 4,096 bytes of Random Access Memory. Each board contains memory protect circuitry, and address selection circuitry for any one of 16 starting address locations in increments of 4K.

Hardware

Dimensions: 10" x 5"

Altair Slots: One

Power (worst case):
+5V at 450 milliamps
+12V at 290 milliamps
+12V at 10 milliamps (unselected)

Epoxy solder mask
Sockets provided for RAMs.

No hardwire jumpers.

A DIP switch is used for board select.

Capabilities

Runs at maximum speed (no wait states).

The entire 4,096 bytes of memory can be protected by switching to PROTECT.

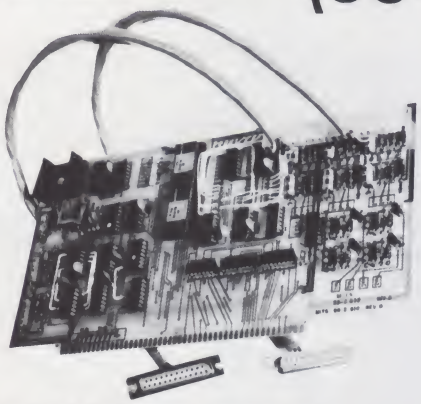
Access Time: 200-300 ns.

Price: $155 kit
$255 assembled

mits

2450 Alamo S.E. / Albuquerque, New Mexico 87106

2-Port Serial I/O Board (88-2SIO)



The 88-2SIO Board design is based upon an Asynchronous Communications Interface Adaptor (ACIA). The ACIA contains both the control register and the status register, so that most options are software selectable.

Hardware

Level Selection: Wire jumpers.

Baud Rate Generator: Crystal-controlled CMOS divider/low power multiplexer.

Device Connection: 12 conductor cable:
10-pin removable connector on board
25-pin connector on back panel

Altair slots: One

Power:
+5V at 520 milliamps
+15V at 70 milliamps
−15V at 70 milliamps

Capabilities

The 88-2SIO board has two serial input/output ports. Each port provides 5 control lines, allowing maximum utilization of sophisticated terminals.

The five control lines are:

Transmit Data, Receive Data, Data Carrier Detect, Clear To Send, Request To Send

All lines are user selectable for ±9v levels (RS-232), TTL levels (0-5V), or 20 milliamp current loop (Teletype).

Full 8-bit Status Register provides:

Received Data Available	Framing Error
Transmitter Buffer Empty	Received Data Overflow
Carrier Detect	Parity Error
Clear to Send	Interrupt Request

The 8-bit status register allows for greater control and handshaking ability.

Provides an on-board, crystal-controlled clock for any of 8 baud rates (with a single jumper):
110, 150, 300, 1200, 1800, 2400, 4800, 9600
A programmable counter can provide other baud rates:
37.5 75 600

2SIO, with 2 ports, can interface 2 serial I/O devices, each running at a different baud rate and each using a different electrical interconnect. Thus, 2 ports can be operating entirely independently of each other (such as an RS-232 CRT terminal running at 9600 baud and a 20 milliamp Teletype running at 110 baud).

Interface levels: TTL, RS-232, TTY.

Bit Configuration: Software selectable for 7 or 8 bits, 1 or 2 stop bits, odd or even parity.

Price: $148 kit **extra port: $38.00**
$180 assembled **extra port: $50.00**

4-Port Parallel I/O Board (88-4PIO)



The 88-4PIO board design is based upon a peripheral interface integrated circuit. The chip contains all control and data registers, thus most 88-4PIO options are software selectable. These options include data direction (each data line can act as input or output) and interrupt/control structure modification. The 88-4PIO board can be expanded to four ports (four 6820 ICs), making it one of the most powerful and versatile interface boards available.

Hardware

The 4PIO board connects to the back panel of the computer with a removable flat cable. The cable has a 24-pin removable plug on the board, and a 25-pin connector on the back panel.

Altair slots: One.

Power: +5 volts at 500 milliamps with 4 ports.

Capabilities

The 4PIO board is capable of handling up to 4 ports on one plug-in card. Each port contains 16 data lines. Each line can be initialized as input or output to interface a terminal (8 lines in, 8 lines out).

Each port on the 4PIO board can handle 2 inputs (such as a paper tape reader and keyboard) or two output devices (such as a paper tape punch and printer) or any combination of custom applications.

All data lines are fully TTL compatible. Eight of the 16 lines are capable of directly driving the base of a transistor switch (1.5V at 1 ma).

A 4PIO with 4 ports has 64 data lines (each group of 8 is individually selectable) and consumes 500 ma at 5V—typical.

Bit Configuration: Each data line can be software selected to act as an input or output.

Price: $105 kit
$130 assembled
for each extra port (up to 3) add:
$38 kit $55 assembled

altair™
Floppy Disk (88-DCDD)

The 88-DCDD is comprised of the Disk Controller and one Disk Drive with an interconnect cable. The Disk Controller consists of 2 PC boards (over 60 ICs) that fit in the Altair chassis. Each controller can address up to 16 disk drives. The Disk Drive unit consists of a PERTEC FD-400, a power supply PC board, and a Buffer/Address/ Line Driver PC board. The Disk Controller converts the serial data to and from 8-bit parallel words (one word every 32 microseconds). The Disk Controller also controls all mechanical functions of the disk as well as presenting disk status to the computer.

Software and System Features

Altair Disk Extended BASIC is an enhanced version of Altair Extended BASIC with added capabilities for saving and loading programs, and for manipulating data files on disk.

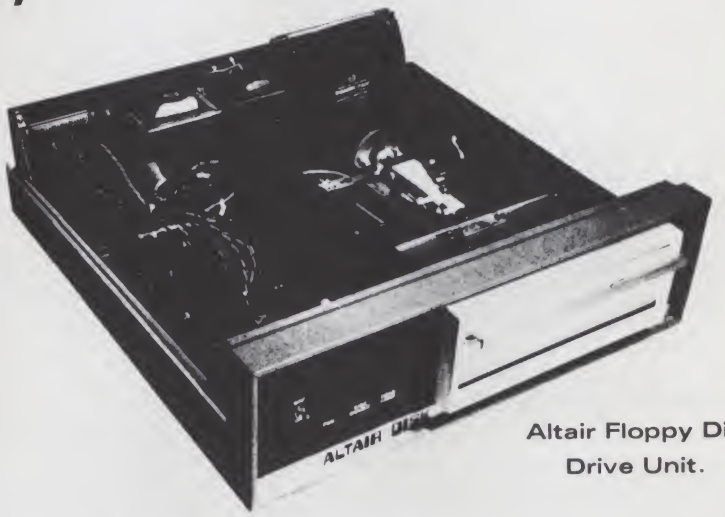
Altair Disk Extended Basic uses random and sequential files for storing information on disk.

Utility software is included with Altair Disk Extended BASIC for copying diskettes, initializing blank diskettes, listing directories, etc.

Disk bootstrap loader is available on paper tape, cassette tape, or PROM (used with 88-PMC PROM Memory Card).

Altair Disk Extended BASIC requires a minimum of 20K of RAM memory for operation.

PROM Disk Bootstrap loader allows loading of Altair Disk Extended BASIC in less than 10 seconds from the time power is turned on.



Altair Floppy Disk Drive Unit.

B. Hardware Specifications

Access Time:
- Track to track: 10 ms.
- Head load and settle time: 45 ms.
- Average time to read or write: 400 ms.
- Worst case: 1135 ms.

Rotational speed: 360 RPM (166.7 ms/rev)

Tracks: 77 per disk

Sectoring: Hard sectored, 32 sectors per track, 5.2 ms/sector (non IBM compatible)

Data Transfer Rate: 250,000 bits/sec. (one 8-bit byte every 32 microseconds)

Maximum number of drives per system: 16

Data storage capacity: 310,000 bytes per disk

Data bytes per sector: 128

Data bytes per track: 4,096

Disk Drive head life: over 10,000 hours of diskette to head contact

Disk Drive MTBF: exceeds 4,000 hours

Disk Drive data reliability: not more than 1 in 10^9 soft (recoverable) errors, 1 in 10^{12} hard (non-recoverable) errors

Power:
- Controller: 1.1 amps at + 8V unregulated (from Altair bus)
- Disk Drive Unit: 110 watts 50/60 Hz 117/220 VAC

Diskette: Hard sectored, 32 sectors + index hole (Dysan #101, ITC #FD 32-100)

Disk Drive Unit Weight: 40 pounds

Price: $1,480 kit
$1,980 assembled

mits

2450 Alamo S.E. / Albuquerque, New Mexico 87106

AUDIO CASSETTE RECORD INTERFACE (88-ACR)

The 88-ACR allows reading and writing of data on audio cassettes, using frequency modulation. The 88-ACR may be used with any medium (or better) quality cassette tape. (Music quality recorders generally give better performance than smaller portable recorders.)

Features

The 88-ACR may be used with machine language Read/Write programs or with ALTAIR BASIC commands, CSAVE and CLOAD, (8K and Extended versions).

ALTAIR software is available on audio cassette tapes for loading into the ALTAIR through the 88-ACR.

Uses any tape recorder with less than 2% wow and flutter. The better the speed stability, the better the data integrity.

Hardware

General Description:

Consists of 88-SIOB (Serial I/O Board) mated to a MODEM (Modulator/ Demodulator) board.

Requires one slot in the ALTAIR bus.

Connects to a tape recorder via miniature phone jacks (user supplies the interconnect cord).

Specifications

Modulation Frequencies: Logic 1-2400 Hz, Logic 0-1850 Hz.

Data Rate: 300 baud, 30 bytes/second.

Bit Format: UART type (1 start bit, 8 data bits, 1 stop bit).

Output Level: 100 MV p-p sawtooth, suitable for "MIC" input of recorders.

Output Impedance: 1.5K ohms.

Input Level: 200 MV p-p to 10 V p-p.

Input Impedance: 100K ohms.

Speed Tolerance: 2% without adjustment of PLL, 5% with adjustment of PLL.

Connector Type: 3.5 mm (⅛ in.) miniature phone jack.

Price: $138 kit
$195 assembled

High Speed Paper Tape Reader (88-HSR)



The High Speed Paper Tape Reader design is based upon the REMEX 300-character-per-second opto-reader. The 88-HSR connects directly to 1 port of an 88-4PIO parallel interface card or 1 parallel port of an Altair 680b computer.

Hardware

Dimensions: 6½" high x 8½" wide x 11" deep.

Drive: DC stepping motor with sprocket drive.

Power: 50 watts, running.

READ Mechanism: Filament lamp to fiber optics to photo cells. The lamp is operated below voltage rating to greatly prolong life.

Tape loading: Easy in-line front load.

Software Features

Start/stop on character.

Low power standby mode. The standby mode reduces motor voltage during periods of inactivity, and allows tapes to be read at the reduced speed of 30 cps.

Specifications

Reading Speed: 30 cps or 300 cps, software selectable, stops "on character."

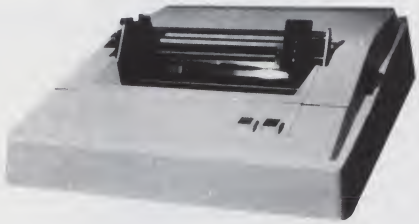
Tapes:
Light transmissivity: 57% or less
Thickness: .0027-.0045 inch
Type: standard 8-track (1-inch) and most other standard 5, 6, or 7 track

Data Output: TTL:

a. Less than .4 volts at 16 ma (no hole); 2.4-5 volts at .2 ma (hole).
b. Plus-in compatible with 88-4PIO or Altair 680 parallel port.

Price: $850.00 Assembled

110 Line Printer (88-LP) and Control Board



The 88-LP produces 80 columns of 5 x 7 dot matrix characters at 110 characters per second (70 lines per minute), and will print up to three copies, plus the original. The universal input transformer allows world-wide use. The Line Printer is fully compatible with LP BASIC (8K or Extended).

The Line Printer Controller (88-LPC Board) provides communication between the Altair and the Line Printer. The 88-LPC can be jumpered for any one of 128 addresses and has full hardware interrupt capability.

Hardware

Dimensions: 18" wide x 8" high x 23" deep.

Weight: 32 lbs.

Requires 1 slot in the Altair bus.

All control electronics are contained on a single 10" x 5" printed circuit board.

Operation

Print Method: Impact dot matrix prints bi-directionally, using a conventional typewriter ribbon.

Print head moves uniformly in both directions and pauses only at the end of a line, minimizing vibration and wear.

Opto-electronic sensing is used to accurately position each dot and to permit characters to be printed on the fly.

Character Set: 64 character subset of ASCII.

Character Spacing: 10 char./inch.

Line Spacing: 6 lines/inch.

Line Width: 80 characters.

Paper Feed: Tractor feed, adjustable 2"-9."

Paper Media: 2"-9," continuous fold.

Line Voltage: 110, 117, 234 VAC ± 10% 50-60' Hz (factory selectable).

Price: $2200 kit
$2375 assembled

Vector Interrupt / Real Time Clock (88-VI/RTC)

The 88-VI gives the computer the capability to interrupt activity via the Restart (RST) instruction and to allow only the highest active priority of the 8 priority levels to interrupt the CPU.

The ENABLE INTERRUPT instruction permits interruption. ENABLE INTERRUPT must also be activated after each interrupt is completed in order to reactivate the CPU's internal interrupt.

The Interrupt Service Handler is the software device which supervises all individual device service routines.

The Real Time Clock provides the computer user with the capability to interrupt the processor at one of four selectable rates. The RTC generates an interrupt after a precise interval of time, which enables software to time certain routines and even to generate the correct time, day, and year, upon request.

The RTC source may be selected from either a derivative of the 2 megahertz clock or the line frequency. The 2 megahertz clock should be used if a fast RTC is needed; it is selectable for time intervals down to 100 microseconds. The line frequency (60 Hertz) should be used in systems that require accuracy over a long period of time.

Specifications

Clock Source Intervals:
System Clock (2 MHz): 100 ns, 1 ms, 10 ms, or 100 ms.

Line Frequency (60Hz) Intervals: 16.67 ms, 166.7 ms, 1.667 sec., or 16.67 sec.

Accuracy: 2 MHz, .01%.

Altair Slots: One

Power: 5 volts at 300 ma.

Price: $138 kit
$185 assembled

mits

2450 Alamo S.E. / Albuquerque, New Mexico 87106

altair T.M. 680b

The ALTAIR 680b microcomputer is an excellent compromise between computer power and low cost structure, without sacrificing design reliability. The system is based on the 6800 microprocessing unit, which adapts nicely to a minimum design configuration. The ALTAIR 680b measures 11-1/16" wide x 11-1/16" deep x 4-11/16" high. The basic system is available in two configurations, depending on the intended application.

Almost all of the 680b circuitry is contained on a single large printed circuit board, including memory and a built-in I/O port. The full front panel model contains all of the controls necessary to program and operate the computer and includes an additional printed circuit board, which provides all of the logic circuitry necessary to reset, halt or start the processor. Also located on this board are switches and associated LED indicator lights for each of the sixteen address lines and eight data lines. The front panel circuit board mounts directly to the main printed circuit board via a 100-contact edge connector. The power switch is located on the back panel of the unit for safety purposes. A "turn-key" front panel model, which eliminates all control except restarting the processor, is also available.

At the heart of the 680b system is the 6800 Microprocessing Unit, which is largely responsible for the overall simplicity of the 680b design. The 6800 MPU contains three 16-bit registers and three 8-bit registers. The program counter is a two byte register which keeps track of the current address of the program. The stack pointer is also a two byte register which keeps track of the current address of the program and contains the next address in an external, variable length push-down/pop-up stack. The index register is a two byte register used to store data or a memory address for indexed addressing operations. There are two single byte accumulators used for holding operands and results from the arithmetic logic unit (ALU). The 8-bit condition code register indicates the results of an ALU operation. In this register there are two unused bits, kept at a logic one. The remaining six bits are used to indicate the status of the following: carry; half carry; overflow; zero; negative; interrupt.

The 6800 has seven different addressing modes, with the particular mode being a function of both the type of instruction and the actual coding within the instruction. The seven modes include the following: Accumulator Addressing—one byte instructions, specifying either of the two accumulators; Immediate Addressing—two or three byte instructions, with the MPU addressing the location given in the 2nd or 2nd and 3rd bytes when the immediate instruction is fetched; Direct Addressing—two byte instructions which allow the user to directly address the lowest 256 bytes of memory in the machine; Extended Addressing—three byte instructions, the second two bytes referring to an absolute address in memory for the operation; Indexed Addressing—two byte instructions, the second byte being added to the 16-bit index register to give the address of the operand; Implied Addressing—one byte instructions and the instruction itself gives the address; Relative Addressing—two byte instructions where the second byte is added to the lower 8 bits, allowing the user to address memory + 129 to -125 bytes from the location of the present instruction.

NEW MEMORY FEATURES

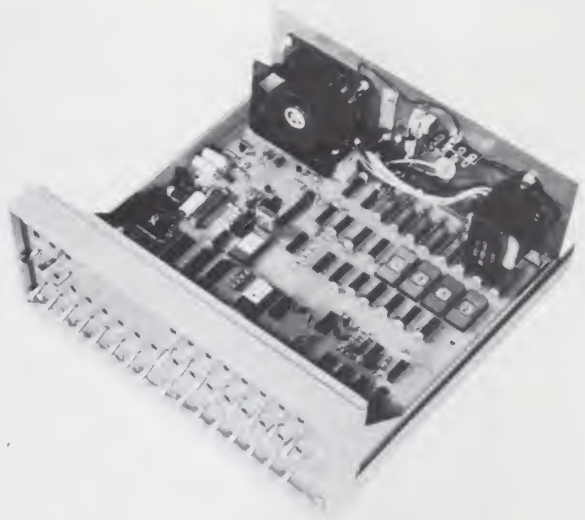
MITS is pleased to announce the development of a 16K static memory card for the Altair 680b. With an access time of 215 nanoseconds and low power consumption of 5 watts, we feel that this is an excellent addition to the Altair 680b.

The 680b cabinet has room for up to three 16K static memory cards, thereby increasing the memory of the Altair 680b to 49K.

SPECIAL FEATURES

ROM monitor.

1702A PROM monitor chip programmed so that you can immediately load and run paper tape object programs such as the text editor and assembler (see below).



Asynchronous Communication Interface Adapter (ACIA).

Allows the machine to transmit and receive a character at a time rather than one bit. Minimizes software needed for I/O routines. Contains crystal clock for baud rate synchronization. User-selectable for RS232, Baudot, TTY, 20ma current loop. Baud rates of 50, 75, 110, 134.5, 150, 200, 300, 600, 1200, 1800, 2400, 4800, and 9600.

Two Pass Resident Assembler and Text Editor

A two pass resident assembler and text editor are available for assembly language programming. This software is compatible with Motorola's format for assembly language programs, text and object files. 8K bytes of memory are required to run this package. The assembler produces a full assembly listing on the second pass, including the hex codes for the location counter and the instruction mnemonics. A symbol table listing is also produced. The text editor has full capabilities for text editing, including line insertion, printing, deletion and modification; as well as commands for changing one string of characters to another and for searching the text buffers for a particular character string.

Basic Interpreter

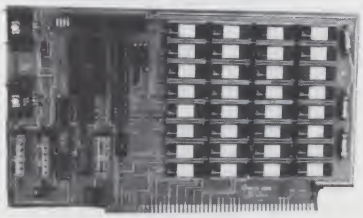
A BASIC interpreter has been developed which will be comparable to the 8800 8K BASIC interpreter.

Altair 680b
Specifications

No. of Boards	Up to 3 additional
Microprocessor	
Model	6800
Technology	NMOS
Data Word Size, Bits	8
Instruction Word Size, Bits	8
Clock Frequency,	500KHz
Add Time, Register to Register, Microsec. Per Data Word	4
Number of Instructions	72
Input/Output Control	
I/O Word Size, Bits	8
Number of I/O channels	256 Memory Address Locations Designated
Interrupt Capability	Std.
Type of Interrupt System	Maskable (Interrupt Request) and Non-maskable Interrupt
Software	
Resident Assembler and Editor	Yes
Higher-level language	BASIC
Monitor	Resident System Monitor on PROM

Price: $466 kit
$625 assembled

altair™ 680b 16K Static Memory Card (680b-BSM)



The 680b-BSM 16K memory card consists of four 4096 x 8 bit static memory arrays, an Address Select circuit, a Read/ Write circuit, and three voltage regulators.

Hardware

A DIP switch is used for address selection (no address jumpers).

Test points have been installed at important signal outputs for ease of checkout and troubleshooting.

Ferrite beads are used on all common supply lines for noise isolation.

Epoxy solder masks cover areas not to be soldered.

Sockets are provided for all ICs.

Power: + 16V (130 milliamps, maximum)
− 16V (110 milliamps, maximum)
+ 9V (150 milliamps, maximum)

Capabilities

Power consumption of 5 watts or 38 microwatts per bit. This allows operation of two 16K cards without adding a second power transformer.

RAM Access Time: 215 ns., max.

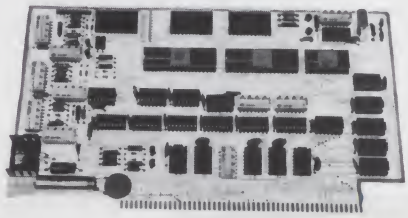
RAM Cycle Time : 400 ns., min.
Up to three 680b-BSM 16K memory cards may be used with the Altair 680b.

Price: $685 kit
$865 assembled
price includes software

mits

2450 Alamo S.E. / Albuquerque, New Mexico 87106

altair™ 680b UNIVERSAL I/O Card



The 680b Universal I/O card provides two parallel ports and one serial port which greatly enhance the I/O capabilities of the Altair 680b. The design of the I/O's two parallel ports is based upon a Peripheral Interface Adapter (PIA). The PIA contains all control and data registers, thus most options are software selectable. These options include data direction (each data line can act as an input or an output) and interrupt/ control structure. The design of the Universal I/O's serial port is based upon an Asynchronous Communications Interface Adapter (ACIA). The ACIA allows serial data to be taken in on its receive line and transfers the data onto the Data Bus, or data can be entered from the Data Bus into the ACIA and then sent out the transmit data line in serial form.

Power: Fully expanded: +9 unreg. at approximately 350 milliamps

TTY: Receive and Transmit lines only: +16v at 44 ma typical
−16v at 20 ma typical

TTY: All input and output lines in use: +16v at 60 ma typical
−16v at 60 ma typical

ACIA not used: +16v at 0 amps
−16v at 0 amps

RS-232 with maximum inputs and outputs: +16v at 0 amps
−16v at 4.8 ma typical

680b MB slots: One.

Now you can buy an Altair™ 8800b or an Altair 680b computer right off the shelf. Altair plug-in boards, peripherals, software and manuals are also available. Check the list below for the MITS dealer in your area.



off the shelf.

RETAIL COMPUTER STORE, INC.
Tim & Susanne Broom
410 NE 72nd St.
SEATTLE, WA 98115
(206) 524-4101

COMPUTER KITS (S. F. area)
Pete Roberts
1044 University Ave.
BERKELEY, CA 94710
(415) 845-5300

THE COMPUTER STORE
(Arrowhead Computer Co.)
Dick Heiser
820 Broadway
SANTA MONICA, CA 90401
(213) 451-0713

GATEWAY ELECTRONICS, INC.
George Mensik OF COLORADO
2839 W. 44th Ave.
DENVER, CO 80211
(303) 458-5444

COMPUTER SHACK
Pete Conner
3120 San Mateo NE
ALBUQUERQUE, NM 87110
(505) 883-8282, 883-8283

ALTAIR COMPUTER CENTER
110 The Annex
5345 East Forty First St.
Tulsa, OK 74135

COMPUTER PRODUCTS UNLIMITED
Harry & Margaret Mohrmann
4216 West 12th
LITTLE ROCK, AR 72204
(501) 666-2839

GATEWAY ELECTRONICS, INC.
Harry & Margaret Mohrmann
Lou Elkins, Stuart Bartfield
8123-25 Page Blvd.
ST. LOUIS, MO 63130
(314) 427-6116

CHICAGO COMPUTER STORE
Lou Van Eperen
517 Talcott Rd.
PARK RIDGE, IL 60068
(312) 823-2388

THE COMPUTER ROOM
3938 Beau D'Rue Drive
Eagan, MN 55122
Dale Hagert, Bob Raemer
(612) 452-2567

BYTE'TRONICS
John & Stan Morrow
Suite 103
1600 Hayes St.
NASHVILLE, TN 37203
(615) 329-1979

THE COMPUTER SYSTEMCENTER
Jim Dunion, Rich Stafford,
Steven Mann, Ron Roberts
3330 Piedmont Road
ATLANTA, GA 30305
(404) 231-1691

THE COMPUTER STORE, INC.
Sid Halligan
120 Cambridge St.
BURLINGTON, MA 01803
(617) 272-8770
Jeff Feldman, Service Dept.

THE COMPUTER STORE OF NEW YORK
Bob Arning
55 West 39th St.
NEW YORK, NEW YORK 10018
(212) 221-1404

THE COMPUTER STORE OF
Peter Blond ANN ARBOR
310 East Washington Street
ANN ARBOR, MI 48104
(313) 995-7616

THE COMPUTER STORE, INC. (Hartford area)
George & Susan Gilpatrick
63 South Main Street
WINDSOR LOCKS, CT 06096
(203) 627-0188

MICROSYSTEMS (*Washington, D.C.*)
Gloria & Russell Banks
6605A Backlick Rd.
SPRINGFIELD, VA 22150
(703) 569-1110

THE COMPUTER STORE
Stephen Payne
1114 Charleston National Plaza
CHARLESTON, W. VA. 25301
(304) 343-4607

MARSH DATA SYSTEMS
Don Marsh
5405 B Southern Comfort Blvd.
TAMPA, FL 33614
(813) 886-9890

mits

Along with pointing out the differences between Tiny BASIC and standard BASIC, Tom offers here some comments and opinions on BASIC and structured programming. Interestingly, his manuscript is one of the few we've received which was prepared using a text editor (a Model 37 TTY driven by a COSMAC 1802 microprocessor). It would seem that more of us (including myself) should be at this stage by now. — John.

Tiny Basic

... a mini-language for your micro

Tom Pittman
PO Box 23189
San Jose CA 95153

If you have an Altair or IMSAI computer or any 8080-based system, you have your choice of several versions of BASIC. There are rumors of BASIC for 6800 and 6502 within the next few months. But these require memory — probably more than you have with your low budget machine.

The alternative is *Tiny BASIC*. The language is a stripped down version of regular BASIC, with integer variables only — no strings, no arrays, and a limited set of statement types. It was first proposed by Bob Albrecht, the "dragon" of Peoples Computer Company (PCC) in Menlo Park, as a language for teaching programming to children. The PCC newspaper ran a series of articles (largely written by Dennis Allison) entitled "Build Your Own BASIC," suggesting how Tiny BASIC might be implemented in a microprocessor. The important portions of these articles have been reprinted in Dr. Dobb's *Journal of Computer Calisthenics and Orthodontia*, published by PCC and available in most computer stores.

BASIC

Before we get into Tiny BASIC, let us look at high level languages in general and BASIC in particular.

When you program in machine language, each command, or statement, represents one operation from the machine's point of view. When we think of a single concept like, "A is the sum of B and C," a machine language program to perform this operation may take several operations, such as:

```
LDA   B
LDA   C
STO   A
```

A high level language, on the other hand, lets you put a single human idea into a single program statement, for instance:

```
LET A = B + C
```

BASIC is one of a class of "algebraic" languages in that it permits the representation of algebraic formulae as part of the language. Other languages in this class are FORTRAN and ALGOL. COBOL does not generally fall in this class (except for the "super" versions).

Of critical importance to all algebraic languages is the concept of an expression. An expression is the programming language notation for what we might think of as "the right-hand side of a formula." Alternatively, we can think of an expression as "a way of expressing the value of some number which the computer is to compute."

An expression may consist of a single number, a single variable name (all variables are referred to by name in high level languages), a single function call (discussed in detail later), or some combination of these, separated by operators and possibly grouped by parentheses. For this discussion, when we refer to an operator, we mean one of the four functions found on a cheap pocket calculator: addition symbolized by " + "; subtraction by " - "; multiplication by " * " (we do not use "X" because that would be confused with the name of the variable "X"); and division by " / ". (The usual symbol for division does not appear on most typewriter and computer keyboards.) Thus,

$$\frac{A-B}{C-D}$$

becomes, in computerese,

```
(A - B) / (C - D)
```

Here the parentheses are used to indicate priority of operations. Normally multiplication and division are performed first, then addition and subtraction. Without the parentheses the expression,

$$\frac{A-B}{C-D}$$

would be understood by the high level language as,

$$a - \frac{B}{C} - d$$



Photo courtesy of Electronic Product Associates, Inc., 1157 Vega Street, San Diego CA 92110.

which is not the same at all.

In BASIC, when an expression is encountered, it is evaluated. That is, the values of the variables are fetched, the numbers are converted (if necessary), the functions are called, and the operations are performed. The evaluation of an expression always results in a number which is defined to be the value of that expression.

The first example which we discussed showed a simple BASIC statement,

```
LET A=B+C
```

This is called an assignment statement, because it assigns the value of the expression "B + C" to the variable A. All algebraic high level languages have some form of assignment statement. They are characterized by the fact that when the computer processes an assignment statement, a single named variable is given a new value. The new value may not necessarily be different from the old; for example:

```
LET A=A
```

This is also a valid assignment statement, even though nothing changes. Assignment statements are also used to put initial values into variables, for instance:

```
LET P=3
```

Control Structures

One of the important characteristics distinguishing different high level languages is the control structure afforded to the programmer. The control structure is determined by the various permitted control statements, which alter the flow of program execution. Normally program execution advances from statement to statement in sequence, although there are however, circumstances in which this sequence is altered. The most common control structure allows one set of operations to be performed if a certain condition is true, and another, if it is false. In "structured programming" this is referred to as the "IF . . . THEN . . . ELSE" construct; its general form is "IF condition is true, THEN do something, ELSE do some other thing." The full generality of this control structure is not directly available in BASIC, but, as we shall see, this is only a minor inconvenience.

Standard BASIC uses the IF . . . THEN construct, and makes it work something like a conditional GOTO:

```
IF A>3 THEN 120
```

If the value of the variable A is greater than three, then (GOTO) line 120, otherwise continue with the next statement in sequence. Actually, the condition to be tested consists of a comparison between two expressions, using any of the comparison operators which are given in Fig. 1.

In each case, if the comparison of the two expressions evaluates as true, the implied GOTO is taken; otherwise the next statement in sequence is executed. In Tiny BASIC the syntax is slightly different. Instead of a statement number, a whole statement follows the THEN part of the IF . . . THEN. The comparison above, in Tiny BASIC, would be:

```
IF A>3 THEN GOTO 120
```

But we could also validly write:

```
IF A<=3 THEN LET A=A+10
```

or some such. Note that this is *not* valid in standard BASIC.

The GOTO construct has been the subject of controversy in the last few years. A strong case has been made for "GOTO-less programming" which uses only certain other control structures to achieve structured programs which are more readable and less

=	Equality (the comparison is true if the two expressions are equal)
>	Greater than
<	Less than
< =	Less or Equal (not Greater)
> =	Greater or Equal
< >	Not Equal

Fig. 1. Comparison Operators.

prone to errors. I believe that both good and incomprehensible programs are possible regardless of the control structures used or not used, but I seem to be in a minority at this time. Suffice to say that BASIC is not conducive to structured programming in the technical sense of the term.

Standard BASIC has one control structure which has been omitted from Tiny BASIC. This is the FOR . . . NEXT loop. Normally, if a program requires some sequence to be performed thirteen times, the following program steps might be used:

```
10 FOR I=1 TO 13
20 ...
30 NEXT I
```

Statement 20 would be executed 13 times, with the variable I containing successively the values, 1, 2, 3 . . . 12, 13. In Tiny BASIC the same operation is a little more verbose:

```
10 LET I=1
20 ...
30 LET I=I+1
40 IF I<=13 THEN GOTO 20
```

but, as you can see, nothing is lost in program capability.

Data Structures

Standard BASIC also has some data structures which have not been carried over into Tiny BASIC. The only data structure in Tiny BASIC is the integer number, which is further limited to 16 binary bits for a value in the range of -32768 to +32767. Compare this precision with the six digit precision in standard BASIC, which also gives you fractional numbers (sometimes called "floating point"). Regular BASIC allows arrays, or variables with multiple values distinguished by "subscripts," and strings, which are variables with text information for values instead of numbers. We will see presently how these deficiencies in Tiny BASIC can be overcome.

Input/Output

Thus far we have said nothing about input and output, how to see the answers the computer has calculated, or how to put in starting values. These needs are accommodated in BASIC by the PRINT and INPUT statements. Numbers are printed (in decimal, for us humans to read) at the user terminal by the PRINT statement:

```
PRINT A, B + C
```

This prints two numbers; the first is the value of the variable A, and the second is the value of the expression B+C. In general, the PRINT statement evaluates and prints expressions. It is perfectly valid to write

```
PRINT 1, 123, 0-0
```

although we know in advance what will be displayed on the terminal. To make our output more readable, BASIC permits the program to print out text labels on the data.

```
PRINT "THE SUM OF 1 + 2 IS", 3 + 2
```

will display the line:

```
THE SUM OF 1 + 2 IS 5
```

To feed new numbers from the terminal to the program the INPUT statement is used.

```
INPUT A, B, C
```

will request three numbers from the input keyboard. The more popular versions of Tiny BASIC have an extra capability here beyond standard BASIC, in that the operator can type in numbers *and* whole expressions. Thus, if in response to the INPUT request above, the operator types

```
1+2, 3*(4+5), B-A
```

the variable A will receive the value 3, B will receive the value 27, and C will receive the value 24 = 27-3. Therefore, a program in Tiny BASIC, which permits no text strings, can display and accept as input limited text information:

```
10 LET Y=1
20 LET N=0
30 PRINT "PLEASE ANSWER Y OR N";
40 INPUT A
50 IF A=Y THEN GOTO 100
60 IF A=N THEN GOTO 120
70 GOTO 30
```

This little program asks for an answer, which should be either the letter "Y" or the letter "N" (or their equivalents, the numbers 1 or 0, respectively). If the operator types anything else, the request is repeated. Obviously, this technique will not work for something like a person's name where any letters of the alphabet in any sequence must be expected, but it is certainly an improvement over no alphabetic input at all.

A generalized text output capability in Tiny BASIC depends on another characteristic peculiar to Tiny BASIC and not shared by standard. That is the fact that the line number in a GOTO or GOSUB statement is not limited to numbers only, but may itself be any valid expression which evaluates to a line number. The program which is shown in Fig. 2 prints A, B, or C, depending on whether the variable N has the value 1, 2, or 3. Note that, if N is out of range, nothing is printed.

The USR Function

What about the fact that there are no arrays? Let us turn to the USR function for a way to store and retrieve blocks of data. The remarks which follow apply only to my version of Tiny BASIC and are unique in that respect.

The USR function is invoked with one, two, or three arguments (expressions separated by commas within the parentheses). The first (or only) argument is evaluated to the binary address of a machine language subroutine somewhere in the computer memory. The USR function does a machine language subroutine call (JSR instruction) to that address. The user is obliged to be sure that there is in fact a subroutine at that address. If there is not, Tiny BASIC (and thus your computer) will execute whatever is there. The second and third arguments, if present, will be loaded into the CPU registers before jumping to this subroutine. On exit, any answer the subroutine produces may be left in the CPU accumulator, and it becomes the value of the function. Two machine language routines are already provided with the BASIC Interpreter; if S is the address of the beginning of the interpreter,

```
USR(S + 20, M)
```

has as its value the byte stored in memory at the address in the variable M (that is, the contents of the second argument is evaluated to a memory address). Also,

```
USR(S + 24, M, B)
```

stores the low order 8 bits of the value of B into the memory location addressed by M. The return value of this function is meaningless.

Consider the standard BASIC program in Fig. 3(a) to input ten numbers and print the largest as compared to the Tiny BASIC program in Fig. 3(b).

I have used this example for two reasons: First, it shows how the USR function may be used to simulate the operation of arrays. Second, it is typical of many of the applications commonly ad-

```
10 IF N>0 THEN IF N<4 THEN GOSUB 20+(N * 10)
20 RETURN
30 PRINT "A"
35 RETURN
40 PRINT "B"
45 RETURN
50 PRINT "C"
55 RETURN
```

Fig. 2. Program to Print A, B, or C, depending on the value of N.

to argue for arrays; however, neither real nor simulated arrays are required for this program! Here is the same program, with no arrays:

```
10 LET I=1
20 LET L=0
30 INPUT V
40 IF L<V THEN LET L=V
50 LET I=I+1
60 IF I<=10 THEN GOTO 30
90 PRINT L
```

Summary

Tiny BASIC is not a super language. But, it also does not require a super computer to run. I've given here only a cursory examination of the power of Tiny BASIC. A full description of Tiny BASIC may be found in the Itty Bitty Computers *Tiny BASIC User's Manual*. This comes with a hex paper tape of the program and is available for $5 from: Itty Bitty Computers, PO Box 23189, San Jose CA 95153.

There are different versions for each of the following systems, so be sure to specify which system you are running:

M6800 with MIKBUG, EXBUG, or home brew (Executes in 0100-08FF); AMI Proto board (Executes in E000-E7FF); SPHERE (Executes in 0200-09FF); 6502 with KIM, TIM or homebrew (Executes in 0200-0AFF); JOLT (Executes in 1000-18FF); APPLE (Executes in 0300-0BFF); KIM-2 4K RAM (executes in 2000-28FF).

Although few people have paper tape systems, we are unable to provide the program on audio cassette. But if you request it, we will supply a hexadecimal listing of the program instead of tape which you can key in and then can save on cassette for future use.

If you have a small 8080 system, there are several widely differing versions of Tiny BASIC in the public domain. Most of them have been published in Dr. Dobb's Journal, which is $10 per year from: People's Computer Company, PO Box 310, Menlo Park CA 94025. This journal has also published a number of games which run in Tiny BASIC.

One final comment. Tiny BASIC was originally conceived as "free software" by the people at PCC. The 6800 and 6502 versions described in this article are not free; they are proprietary and copyrighted. Software is my only source of income, and, if I cannot make it from programs like Tiny BASIC, I won't write them. Please respect the labor of those of us who are trying to make quality software available to you: pay for the programs you use. ■

```
10 FOR I=1 TO 10
20 INPUT V(I)
30 NEXT I
40 LET L=V(1)
50 FOR I=2 TO 10
60 IF L>=V(I) THEN 80
70 LET L=V(I)
80 NEXT I
90 PRINT L
```

```
10 LET I=1
20 INPUT V
25 LET V=USR(S=24,I,V)
30 LET I=I+1
35 IF I<=10 THEN GOTO 20
40 LET L=USR (S+20, 1)
50 LET I=2
60 IF L<USR(S+20,I) THEN LET L=USR(S+20,I)
80 LET I=I+1
90 PRINT L
```

Fig. 3. Programs to input ten numbers and print the largest. (a) Standard BASIC; (b) Tiny BASIC.

Rainbow Computing, Inc.

(formerly Rainbow Enterprises)
10723 White Oak Avenue
Granada Hills, CA 91344
(213) 360-2171

"The computer store featuring software support."

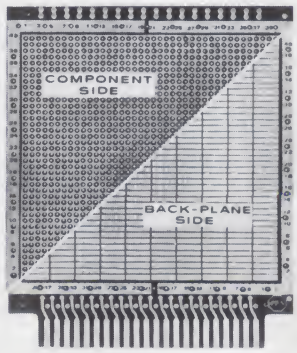
Customized Hardware — Software Packages

Program Conversions & Original Programming

Expert Consulting, Tutoring & Research Services

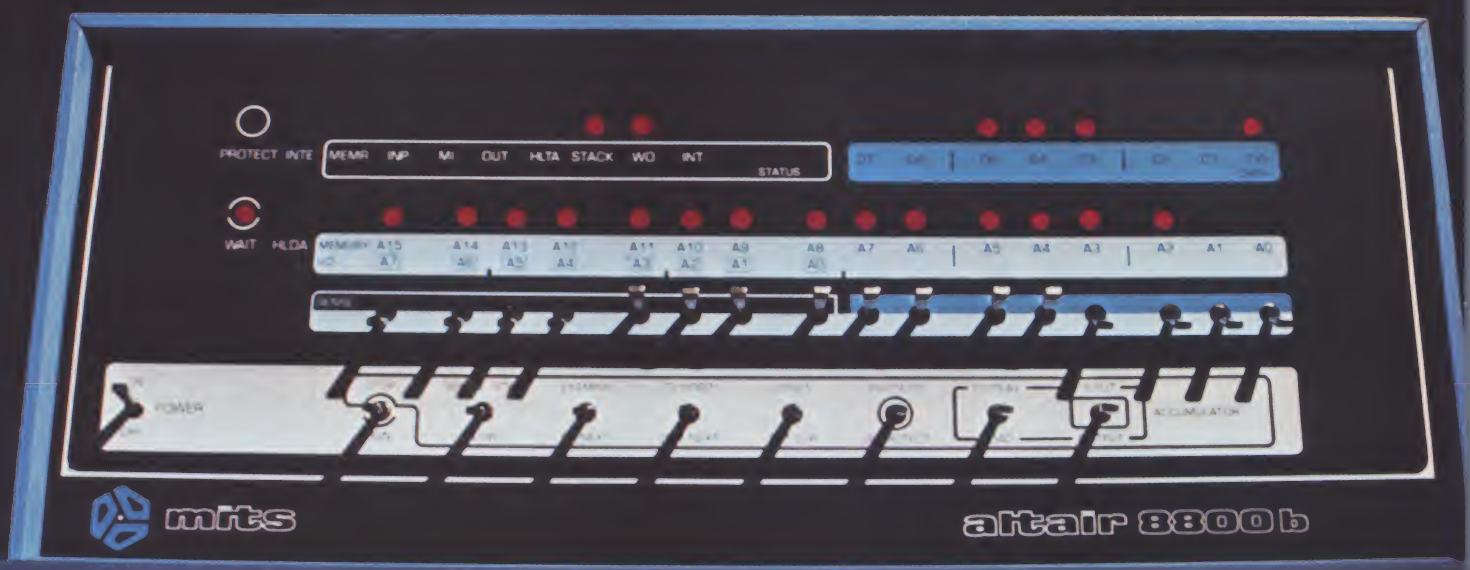
Authorized Distributor for WAVE MATE Microcomputer Systems

ByteBoard-3



4 common connections per backplane pad MINIMIZES/ELIMINATES interconnection wiring 1/16 FR4-tinned.
BB-3 $15.50 ea. $140/doz.

AFS
A. F. STAHLER CO.
P.O. BOX 354
CUPERTINO CA 95014
(408) 252-4219



kilobaud

$7500 SWEEPSTAKES

Why A Sweepstakes?

While many of us get used to seeing mail order sweepstakes contest envelopes (and throwing them out), the fact is that there are very few of us who will not even take a chance on winning an extra computer. And that is what is at stake this time . . . first prize is the new Altair 8800b computer, completely assembled and running with 16K of memory. That would look awfully good on your desk, wouldn't it?

Good Chance To Win!

With over 250 prizes, your chances of winning are a lot better than astronomical . . . after all, the hobby computer field is *still* pretty small. Without an entry in the hat you have *no* chance.

How Does kilobaud Win?

A magazine needs three prime components to succeed . . . readers, a continuing source of good articles to keep the readers happy, and advertisers. In turn, without the readers, the advertisers will go away. It's a great synergistic system. A sweepstakes contest draws attention to the magazine and gets readers to at least try the magazine. From there on the articles have to be good or, as many magazines have found (and are finding) out, the readers will gradually go away . . . and so will the ads.

How Do More Readers Help You?

The more readers a magazine has, the more advertisers it will attract. And the more advertisers, the more pages of articles that can be published. Thus, the more readers you help us get, the fatter the magazine you'll get every month. A recent issue of 73 *Magazine* ran to 288 pages!

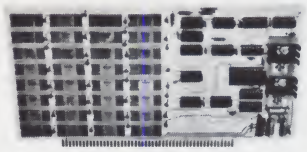
How Can You Help?

Make copies of the Sweepstakes coupon and have friends send them in (hopefully with subscriptions). Make sure anyone with a good article brewing knows that *Kilobaud* is the best payer in the field . . . we want to have the very best articles available. Pass the word at work (if you are around computer oriented people), at school, or around the club. Help us make *Kilobaud* a great big fun magazine.

HERE ARE THE PRIZES!

ALTAIR 8800b

That's right, first prize is a completely assembled and tested Altair 8800b – not a kit. The Altair, the first really good computer available, has become the standard – a recent count showed over 100 different boards designed to plug into it! And software? The 8800b is almost a year ahead of everything else – no system has more!



In addition to the 8800b system you will get one of the brand new MITS 16K memory boards – just what you need to give you plenty of room for extended Basic and workspace. Altair Basic is the one by which others are being measured.

A 460Z

Win the Ohio Scientific CPU Expander, Z80 & IM6100 included. Run PDP8, 8080, Z80 programs on your 6502 (model 400) system.

PROGRAMS

Win a set of 15 game programs for your 6800 by Technical Systems Consultants – enough to keep you busy for a long time.

BOOKS

200 copies of "Hobby Computers Are Here" ($5) will be awarded – this is a fine book to bring newcomers up to speed, particularly in hardware. It is the best selling computer book today.



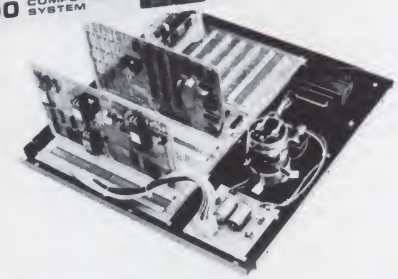
Over 250 PRIZES to be given away!

LIFE SUBSCRIPTIONS

Twenty of these $150 Life Subscriptions to Kilobaud will be awarded during the Sweepstakes contest – this alone is worth sending the card. If you are a subscriber and win a Life Subscription you will get back double your original subscription money.



Win the SWT 6800 computer



with 8K of memory!

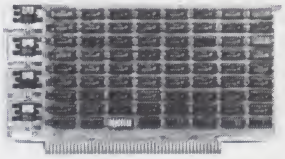
SOUTHWEST TECH 6800

Win a Southwest Technical Products 6800 kit complete with 8K of memory and the new Basic compiler. This is one of the finest 6800 kits you can get – fun to build and a solid performer. The instructions are so good you'll start trying out the parts on the boards and a little while later find yourself making the last solder connections. Thousands of hobbyists are happy with their SWT 6800 computer systems.

With over 250 prizes being awarded in the Kilobaud Sweepstakes, you have to be at the bottom of your luck cycle to miss out completely. Just send in the card and your chances are good. And to make it more interesting, every week till the drawing, we'll pull a card out of the hat for a Life Subscription to Kilobaud. So, the sooner you get your card in the more chances you'll have to win.

SEALS BOARDS

There will be two lucky winners of the Seals 8K "never forget" memory boards – with a keep-alive circuit to protect the memory in case of a power failure or a flicked switch (oops!). Compatible with the Altair and Imsai systems.



Why a contest? We want to get your attention – did you ever see a parade without a big brass band up front? Kilobaud is going to be a big and fun magazine and we want you to read it – and subscribe.

You don't have to subscribe to enter the contest, but we're sure hoping you'll give it a try. Tell you what – if you find that you are willing to do without Kilobaud we'll grudgingly refund the unused part of your subscription (you don't think we're going to be cheerful about it, do you?). You'll like it – our 73 Magazine is enormously popular – you'll like Kilobaud.

Sweepstakes Entry Blank

☐yes! *Enter my*
☐ *One year $15* subscription to Kilobaud.*
☐ *3 year $26* subscription to Kilobaud.*
DRAWING MARCH 15, 1977!

AND make sure my name is in the hat for the Sweepstakes drawings. Start with issue #2.

☐ *NO! . . . No subscription as yet, but put my name in the hat anyway for the Sweepstakes.*
☐ *I'm already a subscriber, enter my name in the Sweepstakes.*

Charge to: ☐ Master Charge ☐ BankAmericard ☐ American Express ☐ Bill Me

Card #_____ Interbank # _____

Expiration date_____ Signature _____

Name _____

Address _____

City _____ State_____ Zip _____

***U.S. & Canada only – others please add $2/year for foreign postage.**

Call TOLL FREE (800) 258-5473 or (800) 251-6771 – SUBSCRIPTIONS ONLY!

poll

I have the following systems running: _____

I am working on the following: _____

I would like to read about – _____

I have so far invested about $ _____

My budget next year is $_____

Would you like an article on – _____

Reo W. Pratt
1370 San Carios Ave.
Concord CA 94518

How a Memory Works

The following article discusses the theory of operation of a particular 4K memory board (Bill Godbout's, to be exact). However, several other manufacturers also offer the same type of static RAM, Altair-compatible boards and the information in this article applies to those boards also. Reo Pratt's reason for selecting the Godbout is rather interesting, too. Read on. — John.

The new microcomputer owner or hobbyist soon learns that a microcomputer main frame is virtually useless for other than basic education without some sort of input and output and more memory than is normally supplied with the average (if there is such a thing) microcomputer kit. This article is about one of the basic "extras" — *memory* — specifically, random access memory or RAM. The objective is to give the novice an understanding of the workings of one of the 4K x 8 RAM board designs currently on the market. The kit chosen for this discussion is the Godbout 4K x 8 Econoram. I chose this kit for several reasons, not the least of which is that I work for Bill Godbout. As a consequence, I'm very familiar with this particular kit. (Besides, Bill would have my head if I wrote about someone else's!) I'm not going to try to evaluate the Econoram's performance or compare it (I admit prejudice), but merely explain its operation.

In covering this subject, I'm assuming some knowledge and experience on the part of the reader. Hopefully, this discussion will help the neophyte to understand the workings of a memory, but, to avoid any long dissertations on basics, I'm assuming a familiarity with the binary number system and the basic gates: AND, OR, NOR and NAND.

The most popular microprocessor chips currently on the market all use an address format of 16 parallel bits. Simple math tells us these CPUs are capable of selecting, counting or addressing $65{,}536_{10}$ separate memory locations (2^{16} = 65,536). This 65,536 is what would commonly be referred to as "64K of memory." Unfortunately, this much RAM is rarely used by most hobbyists because of the expense.

For purposes of definition, the addressable space of a microcomputer is usually divided up into smaller blocks called "pages." A page is an arbitrary dimension and its size is determined by the design of available components and the preference of the user. For this discussion we'll divide our 64K into 16 different blocks of 4K each. The easiest way to see this is to imagine a square divided into fourths (one fourth being a "page"), with each fourth divided into four again. Each of these latter fourths will be one of our 4K x 8 RAM boards.

Let's take a look at the physical layout of a 4K RAM board (Fig. 1). The board can be divided into three parts: First, at the top, are three voltage regulators; then, there are 32 1K RAMs, in this case 2102s; and finally, there are 10 logic support and buffer chips.

The voltage regulators provide on-board regulation. Common microcomputer power supplies produce 8 V, but most TTL and memory chips require 5 V for operation. These voltage regulators give a constant 5 V at their output with anywhere from 7 to 35 V at the input.

Notice that there are four rows of eight 2102s. Each 2102 will store 1,024 (1K) single bits. The memory is organized in the following manners: Each row is eight chips wide, with 1,024 8-bit words per row, and, since there are four rows, we have a

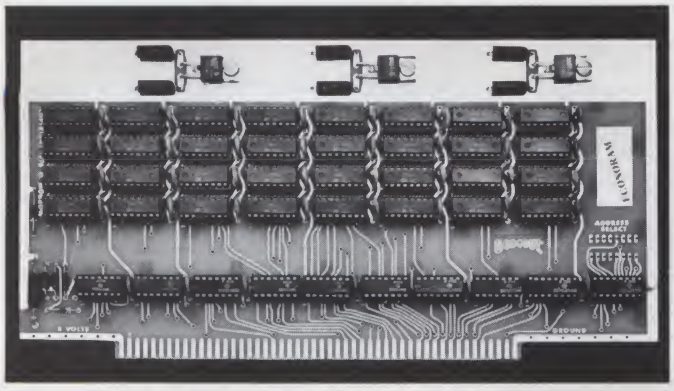


Fig. 1. Component layout of the Econoram *board.*

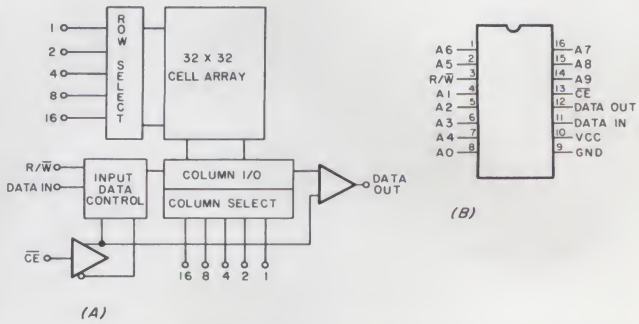


Fig. 2. (a) Block diagram of a 2102 memory chip. (b) 2102 pinout diagram.

4K x 8 memory. It would be possible to take the same number of chips and reorganize them into 8K x 4, for a 4-bit CPU, or 2K x 16 for a 16-bit CPU.

Before we can view the operation of the entire board we need to dig into the depths of the 2102 static random access memory itself. Fig. 2(a) is a logic diagram of a single 2102, and Fig. 2(b) is the pinout. The memory section of the chip itself is organized on the column/row principle. This gives us a grid wherein we may select a particular location (single bit) by simply calling out the column and row. This is illustrated in Fig. 3. For a memory size of 1,024 we'll need 32 columns and 32 rows (0-31). Our column/row selection circuitry will consist of five input bits each ($2^5 = 32$). Let's suppose that we wish to select the memory location marked by the dot in column 2, row 5. We would present the number 2 (00010_2) to the column select logic and the number 5 (00101_2) to the row select logic. The selected bit would then be outputted to the data line of the chip. Just by selecting the desired location with only 10 address bits, we can access 1,024 different memory locations which can be written into or read out of at will. By putting eight of these in a row (bank) and selecting the same location in each chip simultaneously, we can have our choice of 1,024 8-bit words. This would be a 1K x 8 memory.

Referring to the 2102 pinout diagram in Fig. 2(b), we see that the 2102 comes in a 16-pin package. We have now covered 10 of these pins, the address lines. The other 6 are Vcc, ground, data out, read/write (which translates as read not write), and chip enable. Vcc is simply the pin that supplies power to the memory cells and logic arrays of the chip, and ground is self-explanatory. Data in, data out, and the R/$\overline{W}$ line will be discussed together. R/$\overline{W}$ is the pin which determines whether the memory location selected is to have data written into it or read out of it. If the pin is held high (logic 1), the chip will output selected data to the data out line (i.e., a READ operation). If this pin is held low (logic 0) the chip will look for data at the data-in line to write into the selected memory location. Therefore, either a read operation or a write operation will always be taking place, depending on the state of the R/W line.

Pin 13, the chip enable input ($\overline{CE}$), is Tri-state* logic. Chip enable is an active low signal (denoted by the bar over the term $\overline{CE}$) and is used to enable an entire 1K bank of 2102s. The Tri-state operation can be explained as follows: With chip enable high (inactive or disabled), the data out pin (which is tied to the data bus) is in a high impedance state. This provides for good isolation. (The high impedance state is one of the three states.) When the chip enable signal goes low, the data out line will be either a one or a zero, depending on the state of the addressed location. (And, of course, these are the other two states.)

The reason for wanting to disable the outputs is this: Suppose we have four banks (rows) of memory, and we want to read one word from one of the banks. All banks have the same bus or data lines in common. When we present our address to the chips, remember that each chip uses the same address lines. Therefore when we address the four 1K banks, instead of having only one bank place data onto the eight-bit bus, we have four banks. This causes the outputs to "fight" each other, so we must have some way of turning three of the four rows off while we're asking the other one row for data. Chip enable accomplishes this. By disabling all the chips except the ones in the row we wish to access, we will have only one set of memories placing data on the bus at any moment. The enabling and disabling of the desired chips at the right time is accomplished by addressing logic external to the memory chips. Chip enable is an important part of the address structure of the entire board (more about that later).

*Tri-state is a registered trademark of National Semiconductor.

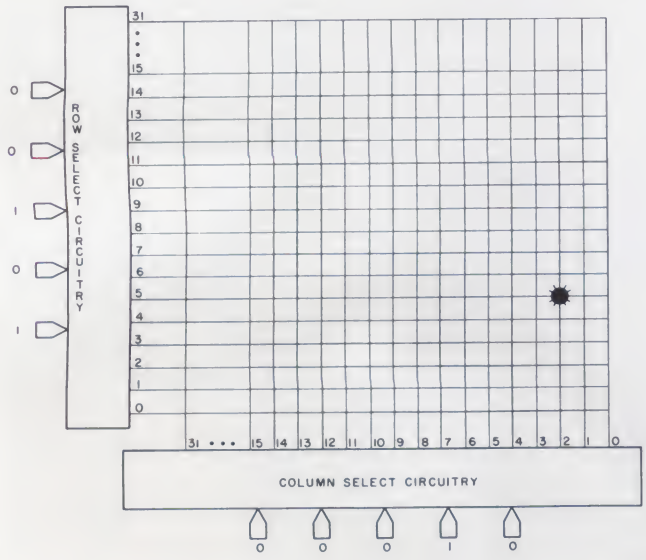


Fig. 3. Memory matrix, illustrating how 1024 different memory locations may be addressed by ten binary address bits.

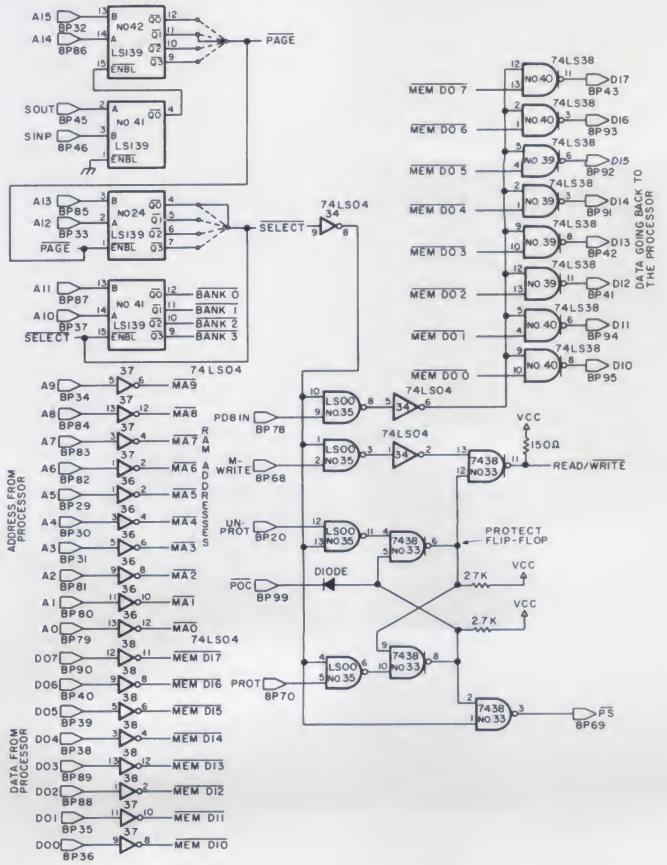


Fig. 4. Schematic of Address, Data in/Data out, and Memory Protect logic.

Having studied the 2102, we're now ready to look at a schematic of the entire board (refer to Fig. 4 and 5). There are basically three parts to a RAM memory; *Addressing, data in/data out,* and *read/write*. Fig. 4 is a schematic of the board addressing and data in/data out logic. In the lower left corner are the data out lines, labeled DO0 through DO7. Notice that signals coming into the board from the processor are denoted by the pointed box. DO1 stands for data out, bit 1. BP 35 simply means backplane, pin 35. Beginning immediately above DO7 on the left side of the page are the address input lines from the processor. In the right field of Fig. 4 are the data output pins from the memory to the processor. The lower center field is the memory protect logic.

Notice that address bits A12 through A15 all go directly to two 74LS139s. These 75LS139s contain the logic for page and board select. The "handling" of these address bits will determine where our board is in memory. Remember that we have divided the total addressable memory of the processor into 16 parts (sixteen 4K segments). By the use of two jumpers or switches on the board, the user selects the board's location in memory. A 74LS139 is made up of two separate two-line to four-line decoders in a single package. A two-line to four-line is a logic array which selects one of four outputs when given a two bit input. Fig. 5 not only illustrates the breakdown of the address lines into BANK, BOARD, and PAGE selection, it also serves as a truth table for the 74LS139.

Suppose we have address bits A15 and A14 both equal to 01. This will cause the outputs at pins 9, 10 and 12 to be HIGH, and that at pin 11 to be LOW. Having pin 11 jumpered to PAGE would place this board in the 16 to 32K addressing range. (PAGE indicates active low.) This signal is then jumpered to enable pin 1 of chip 42, which brings us to address bits A12 and A13. The same logic follows: If inputs at A12 and A13 are, for example, 00, pin 4 of chip 42 would be LOW which would cause bank SELECT to be enabled. Address bits 10 and 11 would then select a particular 1K bank.

If we assume we now have data at the inputs waiting to be stored and we've addressed the board properly, all we need do is make read/$\overline{\text{write}}$ LOW and our data will be written into the addressed location. The bank select outputs of chip 41 go directly to the chip enable pins (pin 13 of the 2102s) in their respective banks. With pin 4 of chip 42 held LOW, SELECT at pin 9 of chip 34 will be LOW, causing the output of the gate to be HIGH, which in turn will enable the inputs of the 74LS00 NAND gates. Then when the processor makes memory write (MWRITE-BP68) high, the output of chip 33 pin 11 will be low, causing the data to be written into the addressed memory location.

Reading the data stored in memory is very simple now that we understand how it got there. We must present an address to inputs A0 through A15 which will cause the same page, board, bank, and chip location to be selected. MWRITE (BP68) will be brought LOW by the processor, which will make R/W high forcing all selected memories to output data. The eight selected memories will output their data on the MEMDO0 through MEMDO7 lines. The 74LS38 output buffer will be enabled by bringing PDBIN HIGH (Processor Data Bus In; BP78, chip 35 pin 9).

The last part of the board which requires discussion is the memory protect circuitry shown in the lower center of Fig. 5. This is simply a buffered flip-flop, the status of which is determined by the memory protect and unprotect lines at BP70 and BP20. If pin 12 of IC33 is held low, the read/$\overline{\text{write}}$ line will be high so that data cannot be written in any memory location on the board. This is the "protect" status of the memory.

I've described here a memory board with buffered inputs and outputs, a memory protect feature and onboard regulation. There are, of course, others available with slight variations, other features, and different methods of doing the same thing, but this discussion should give you a good idea of what's going on in this type of memory. Good luck, and happy remembering. ■

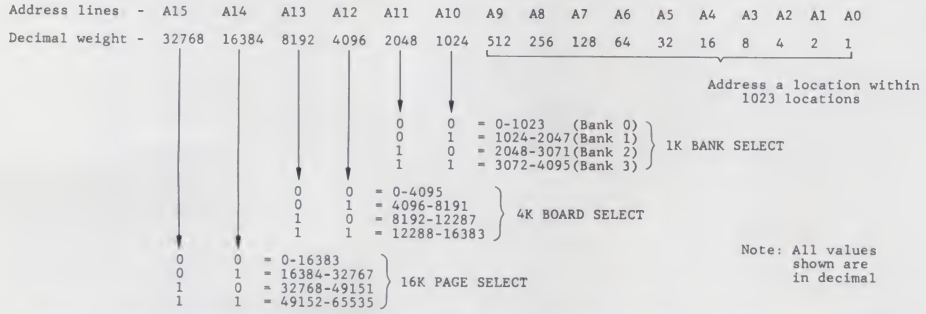


Fig. 5. Breakdown of address lines into BANK, MODULE (Board), and PAGE select functions.

NEWS FLASH!

HERE COMES COMPUTER SHACK

Kilobaud recently interviewed Mr. John Martin, Director of Franchise Sales for *Computer Shack, Inc.* of San Leandro, California. In fact, we interviewed him so recently that you will have to wait for next month's issue to get full details.

Anyway, for right now, here are some of the things talked about: Computer Shacks are on the way . . . no doubt about it. Mr. Martin and his associates (formerly with IMSAI and now colocated with them . . . but separate and distinct corporations) have been negotiating with groups and individuals all over the United States concerning Computer Shack franchises.

Mr. Martin predicted that by the end of 1977 there would be between 100 and 200 Computer Shacks throughout the country. And that means prices for components and systems should drop even lower . . . due to the purchasing power of a group as large as Computer Shack.

The pilot store is located in Hayward, California, and Mr. Martin said that it was quite successful. So, here comes your chance Mr. Hobbyist . . . if you have the initial investment required (ranging from $15,000 to $62,500 depending on location), Computer Shack will take care of all . . . and I mean all . . . the details of starting your very own store . . . training . . . location . . . advertising . . . promotional materials . . . interior design . . . the whole works . . . everything you need to open your own Computer Shack. We'll get details to you in the next issue of *Kilobaud.*

Bob Leach

WHO DARED TO CHALLENGE THE 8080 & 6800 SYSTEMS?

WE DID!



7" x 16" x 14-3/4"

VERAS SYSTEMS "F-8"

A COMPLETE SYSTEM

FOR ONLY $459.00 KIT FORM

$709.00 ASSEMBLED and TESTED

COMPARE STANDARD FEATURES ON BASIC SYSTEMS YOU'LL SEE WHY THE VERAS F-8 SYSTEM WINS HANDS DOWN!

*Registered Trademarks of Fairchild and Motorola

STANDARD FEATURES	VERAS F8	8080 & 8080A SYSTEMS	6800 SYSTEM
Parallel I/O Ports	(3) 8 Bit Bidirectional Brought Out To Rear Panel Conn's.	None	None
Serial I/O Port	RS-232 or TTY Brought Out To Rear Panel Term. Strip.	None	RS-232 or TTY
Interval Timers	(2) Programmable Interval Timers	None	None
Interrupts	a) Vectored Interrupt To Location 0090 Hex. b) Vectored Interrupt Programmable Location c) Two Vectored Interrupts Associated With Interval Timers d) Total of (4) Interrupts In A User Defined Priority Interrupt Structure	None	a) 2 Non Vectored Interrupts on P I A b) 2 Vectored S W I & N M I c) Total of 4 Non Prioritized Interrupts
Built In Mini Operating System in ROM For Terminal And Memory Debug	FAIRBUG*	None	MIKBUG*
Loader Program	Automatic Internal ROM	Manual Console Switches	Automatic Internal ROM
Static RAM Memory	1024 BYTES	None	2048 BYTES
Card Rack	Rugged Alum. Self Contained Card Rack/Plastic Self Aligning Card Guides	Card Supports	None
Auxiliary DC Power To Power Peripherals	+5V, -5V, +12V, -12V @ 1 Amp. Ea. Regulated At Rear Panel Terminal Strip	None	None
Basic Kit Price	$459.00	$539.00, $599.00 or $840.00 Depending On System	$395.00

OUR 4K STATIC RAM BOARD FEATURES: (OPTIONAL)

- On board decoding for any four of 64 pages.
- Address and data lines are fully buffered.
- No onboard regulators to cause heat problems. (Chassis mounted)
- 4K memory boards with connector, buffers, static RAM's & sockets are available in kit form **$149.00**

THE VERAS SYSTEM

The VERAS System can be made into a 17K processor by merely adding four of our optional memory boards. The kit includes everything you need to build the VERAS F-8 Computer as described. All boards, connectors, switches, discrete components, power supply and cabinet are supplied. Programming manual, data book and simplified support documentation supplied, 8K Assembler and Editor (paper tape or K.C. std. cassette) available on request with minimum order of 8K RAM.

The More Flexible and Expandable Computer at a Comparative Price.

VERAS SYSTEMS

Warranty: 90 days on parts and labor for assembled units. 90 days on parts for kits. Prices, specifications, and delivery subject to change without notice.

THE CPU BOARD FEATURES:

- Two I/O ports each on the CPU and ROM chip make 32 bidirectional TTI lines.
- The Fairbug* programmed storage unit provides the programmer with all I/O subroutines, allows the programmer to alter or display memory, and register its contents via teletype.
- Programmable internal timer is built into the ROM chip.
- Built in clock generator and power on reset are built into the CPU chip.
- There is a local interrupt with automatic address vector
- It is expandable to 65K bytes of memory
- 20 mil loop and/or RS232 interface included.
- 1K of on board 2102 RAM.
- Serial interface built into PSU chip

*Fairbug is a registered trademark of Fairchild Corp.

TINY 2 K BASIC (AVAILABLE) OCT. 15, 1976 $25.00

FULL BASIC (AVAILABLE) DEC. 15, 1976 $50.00

Computer dealers and hobbyist club inquiries are invited. Expected delivery time 30 days or less.

VERAS SYSTEMS
A Div. of Solid State Sales, Inc.
Box 74 Somerville, MA 02143
(617) 547-1461

☐ Enclosed is check for $ _____

or ☐ Master Charge # _____

☐ VERAS F-8 Computer Kit ☐ Assembled

☐ 4K RAM Board Quantity _____

Name _____

Address _____

City, State _____ Zip _____

When Art Childs was editor of SCCS Interface Magazine *one of the things he had high hopes for was a software exchange library within the Southern California Computer Society (on a nationwide basis). Unfortunately, the idea never got off the ground. When I told him about the* Kilobaud *Software Library, his eyes did seem to light up a bit. At the end of our discussion we concluded that an article dealing with the criteria for submitting assembly-language programs would be in order. Naturally, we decided Art should be the one to write it. Here 'tis. — John.*

Software Exchange

... smoothing the rocky road

Art Childs
PO Box 430
Glendale CA 91206

Exchange of software among computer hobbyists may be the single most important contribution we can make toward promoting the utility and pleasure of owning a personal computing system. The easier, or more convenient, it is for the hobbyist to contribute, distribute and use software, the greater will be the enhancement of our computing capability. Toward making shared software easier to use, I would like to suggest a few ideas and propose possible standards designed to solve some of the problems attendant to shared software.

Few of us have time to write all the software we might need, and it's unlikely that any of us can gain the expertise in all areas of programming we would need to make our own systems as versatile as we would like. We must each specialize in one or two areas, then share the results of our efforts with those specializing in other areas. With the help of punched paper tape, cassette or floppy disk for media, and the assistance of the U.S. Postal Service for exchange, it's quite likely that within a short time we could each have a large drawer full of useful software. At that point however, we would have considerable work to do to make all that software work for our particular application. In this article, I will examine the obstacles to software exchange and propose some solutions.

Communication

The first and most pressing need is communication. We need to determine who has what, how to obtain it, and at what cost. Many clubs have informal software exchanges. Members can attend monthly meetings, bring the results of their latest programming endeavors to the club pool, and select for their own use the contributions of other members. Although this method of distribution works fairly well, it has its limitations, the most significant of which is determining what is available.

Another limitation is the relatively small size of the group involved in the exchange effort. The active members in a club might number in the hundreds and yet represent only one or two percent of the total number of hobbyists nationwide who would like to participate in a software exchange program.

Yet another problem is the de facto standards that tend to become regional in nature. In the Los Angeles area the prime medium of exchange has become Tarbell cassette, using the same console control mechanisms utilized in most Altair BASIC interpreters, i.e.,

Console Status Port = 00
Console Data Port = 01
Console Input Ready = Bit 0, True Low
Console Output Ready = Bit 7, True Low

In other areas the medium of exchange ranges from punched paper tape object formats (Intel hex object format, binary, baudot, etc.) to cassette in one of many formats and baud rates available, to ICOM floppy disk format.

One result of the lack of communications and standardization is a frustration I am currently feeling. I have in my possession a source listing of the Livermore Labs Basic and would like to share it with all who want it — but, besides finding the time to enter the code and assemble it, how do I inform my fellow hobbyists it is available? What medium of exchange do I select for distribution; punched paper tape, Tarbell cassette, KC standard cassette, and if cassette, what baud rate? Or if I use floppy disk, should it be a soft-sector, IBM compatible format (such as ICOM), or a hard-sector format (such as the Altair), or . . . ?

The Communications Solution

As this is being written, plans are being made at *Kilo-*

baud Magazine by publisher Wayne Green and editor John Craig to establish a software exchange library for the personal computing community. To my knowledge, this is one of the first attempts at undertaking such an endeavor by an organization with the communications resources necessary to accomplish the task. I therefore propose the following as a solution to the communications problem of software exchange:

Given that *Kilobaud* is, or will probably become, one of the outstanding national personal computing publications, the inclusion of a software catalog in each issue informing readers of the latest additions to the software exchange library is recommended. This catalog should include the following for each program listed:

1. Program name
2. Author, author's address (with permission)
3. Very brief description of program
4. Machine written for (8080, 6800, 6502, etc.)
5. Language written in (Assembly, BASIC, FORTRAN, PLM, etc.) If BASIC or other high level language, which version?
6. Memory requirements
7. Peripheral equipment requirements (cassette, floppy, line printer, CRT, etc.) other than console device.
8. Media available on (punched paper tape, floppy, etc.)

Assuming that, like myself, you save all your back issues of technical publications (furniture moving companies love us) you will always have at your fingertips an index to the *Kilobaud* Software Library, and at a glance, know whether a particular program is useful to you on your system.

Source Listing

A comment here regarding the necessity of sources listing is appropriate. If the program is written in BASIC or another interpreter, the code itself is the source listing. If on the other hand, assembly language or a compiler is used, source and object become separate entities and provide two possibilities for the user. If you have the compiler required, the source is all you need. If, however, you don't possess the compiler — or an assembler, you could still use the object code if certain criteria were met when the program was written. These criteria will be discussed later.

In any case, the source code is going to be needed, in almost every case, by users of programs from the library. Therefore, the source code should certainly be submitted with a program and made available to the purchaser. In light of this, a ninth item might be added to the list of catalog data:

9. Type code available (source, object, or both).

Documentation

The next problem is: how do we use the particular program in question? What peripherals does it require? Does it require operator response, and, if so, what kind of response does it expect? Does it want its numeric input to hex, octal, or decimal? How much memory does it require? The answers to these problems lie in what is purported to separate the programmer from the coder — *documentation!*

Before launching into detail, let me clarify a rather important point. Most of the standards suggested here apply to programs written in assembly language, since this is the bulk of my experience on 8080-based computers. Other than modifying BASIC interpreters for friends, I have done little more than play Star Trek with, much less written programs in, BASIC. Thus many of the standards will appear to be software system specifications of a sort.

Documentation is a bothersome, but necessary, evil. The following suggestions for documentation, when used in conjunction with the coding standards listed below, will provide all the information necessary to use the program it describes and, at the same time, hold the length of documentation to something less than a Hemingway novel.

1. *Description* — A brief description of what the program does and, optionally, how it does it.
2. *Language* — The language program was written in (i.e., 8080, 6800, 6502, etc.).
3. *Memory Requirement* — Expressed in bytes.
4. *Origin Address* — Address in memory where the program commences loading *and* the address of any non-contiguous memory requirements. Also, whether the program is relocatable (and therefore provided with a relocating loader).
5. *User Interaction Required* — Those steps the user must take to run the program. For instance, will the machine ask questions on the console device to which the answer may not be apparent?
6. *Complete Description* of any routines called by the program which are not included, such as standard monitor routines; hex-to-decimal, ASCII-to-hex, etc.

Several items are obviously missing from this list. Those items comprise the coding standards mentioned previously. If we adhere to the following when first writing the program, the documentation will be shorter, and we will all be able to get back to writing our next programs that much sooner.

Input/Output

To assume that all systems use an 80 character-per-line CRT for a console would be limiting. Even if that were true, should we expect that all CRTs are addressed to port X and are they all interrupt driven? Obviously not. So how is this problem solved, and what are the attendant difficulties of handling line printer output, paper punch output, or cassette input and output?

One solution might be to disassemble the program in question, analyze the code for input and output instructions, and either modify the code and reassemble or patch the object code to fit our particular system of hardware. This is a workable and educational procedure. But it is time consuming. Further, *good* disassemblers written to run on the object machine are rare. (I had to write my own in order to have a truly useful tool.)

Another method would involve loading the program into your machine and then, using an editor program, search through the program for I/O instructions. Those instructions could then be modified to conform to your particular system. Once again, this is time consuming and difficult.

A more practical solution is to standardize I/O vectors. For those who are unfamiliar with that term, a vector as used in this context would be analogous to a traffic dispatcher or perhaps to a road map. If you want to drive from point A to point B, you

```
        ORG ADDR1
BEGIN:  JMP START
CI:     JMP ADDR2       ;Console input vector
CO:     JMP ADDR3       ;Console output vector
RI:     JMP ADDR4       ;Tape reader input vector
PO:     JMP ADDR5       ;Punch output vector
LO:     JMP ADDR6       ;Line printer output vector
KI:     JMP ADDR7       ;Cassette input vector
KO:     JMP ADDR8       ;Cassette output vector
FI:     JMP ADDR9       ;Floppy input vector
FO:     JMP ADDRA       ;Floppy output vector
A1:     JMP ADDRB       ;Auxiliary or special purpose vectors
A2:     JMP ADDRC       ;
A3:     JMP ADDRD       ;
```

Program (A)

```
          ORG ADDR1
BEGIN:    JMP START
CI:       JMP ADDR2    ;Console input vector
CO:       JMP ADDR3    ;Console output vector
RI:       JMP ADDR4    ;Tape reader input vector
PO:       JMP ADDR5    ;Punch output vector
LO:       JMP ADDR6    ;Line printer output vector
KI:       JMP ADDR7    ;Cassette input vector
KO:       JMP ADDR8    ;Cassette output vector
FI:       JMP ADDR9    ;Floppy input vector
FO:       JMP ADDRA    ;Floppy output vector
A1:       JMP ADDRB    ;Auxiliary or special purpose vector
A2:       JMP ADDRC    ;Auxiliary or special purpose vector
A3:       JMP ADDRD    ;Auxiliary or special purpose vector
CPMSG:    JMP ADDRE    ;Console message output routine
ATOH:     JMP ADDRF    ;ASCII to hex conversion routine-2 characters
NBL:      JMP ADDR10   ;ASCII to hex conversion - single character
HTDA:     JMP ADDR11   ;Hex to ASCII conversion routine
CONV:     JMP ADDR12   ;Hex to ASCII - Single Character
```

Program (B)

```
CPMSG:    MOV B,M      ;Move number of bytes in msg into B (char counter)
MSGA:     INX H        ;Incr H&L to point to 1st char in msg
          MOV C,M      ;Move char into C reg for output
          CALL CO      ;Call console output routine
          DCR  B       ;Decrement character counter
          RZ           ;Return to main program if char counter = zero
          JMP MSGA     ;Jump back to output another character
MSG1:     DB 17        ;Size of message (number of characters)
          DB:          "This is a message"
```

Program (C)

```
          MSG1    DB    17  11
                  DB        54 48 49 53 20 49 53 20 41 20
                            4D 45 53 53 41 47 45
```

Program (D)

```
          ORG ADDR1
ATOH:     MOV  A,B     ;Get high order character
          CALL NBL     ;Convert to hex
          RC           ;Invalid (Return if carry set)
          RLC          ;Rotate to upper nibble
          RLC
          RLC
          RLC
          MOV  B,A     ;Save
          MOV  A,C     ;Get low order character
          CALL NBL     ;Convert to hex
          RC           ;Invalid
          ORA  B       ;Combine with high order and clear carry
          RET
NBL:      SUI  30H     ;Less than ASCII "0"?
          RC           ;Yes--Invalid
          ADI  0E9H    ;Greater than ASCII "F"?
          RC           ;Yes--Invalid
          ADI  6       ;Char=A-F?
          JP   NBLA    ;Yes
          ADI  7       ;Char=(3A)H through 40H
          RC           ;Yes--invalid
NBLA:     ADI  10      ;Adjust
          ORA  A       ;Clear carry bit
          RET
```

Program (E)

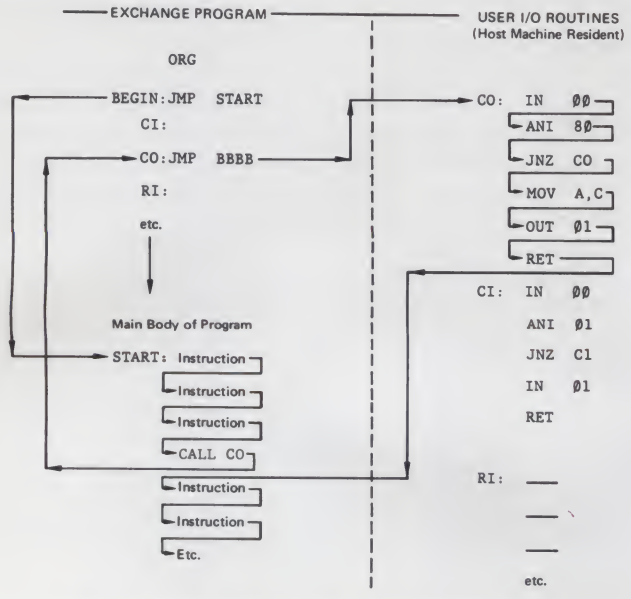


Fig. 1. Exchanged and user program interface.

simply follow the road on the map to point B. A software I/O vector helps a program arrive at the point that will accomplish the particular task required. The mechanism is quite simple.

Assuming each person knows his own machine and has already written the necessary software to handle his I/O devices, all that is necessary is to direct the program to the appropriate I/O routine. This is done with a jump instruction which causes the program to execute the I/O code written for the host machine. For instance, when programmers want to put out a character on the console, they load the output character into the C register in hex ASCII representation and call location X. Users, knowing that the output character will be in ASCII in the C register (for 8080 machines), write their console output routine accordingly. Also, knowing the program will look for the required routine at location X, they will put a jump instruction at location X which will vector (point) the program to the output (or input) routine. Further, if all I/O routines are terminated with a return instruction, the writer needs only to "CALL" location X. All that remains is to define X.

Suppose the first 14 lines of code in all programs were written as in Program (A).

Because documentation accompanying the program gives the starting address, the recipient of the program now makes a few simple modifications which allow the program to run on his system. At location START plus 4, he lays in the address of the console input routine which, hopefully, resides in ROM or PROM or is at least loaded to the same address each time he brings up his system. At location START plus 7, the address of his console output routine is loaded; at START plus 10, the address of his paper tape reader input routine is loaded, etc. (See Fig. 1). The ideal, of course, would be for everyone to locate his I/O routines at the same location, but computer people, being highly individual creatures, will not likely accomplish that soon, if ever.

For this standard to work, we must add a couple of attendant rules. First, an output character must always be handed to the output routine in the C register. It could just as well be the B register, or the D or E, for that matter, but the D and E are more useful as program indexes than are B and C, so using the C register is less disruptive to good coding technique. Also, the C register has become a de facto standard in many areas. Many of Intel's programs, for instance, are coded that way. There are other techniques. However, everything considered, using the C register seems to be the best option (and is therefore my proposed standard).

The second rule is that input data will always be returned in the A register. The reason is because the data input from a port appears in the A register and, hence, wouldn't require any further manipulation by the input routine.

External Routines

External routines usually refers to code not included in the program. This code generally includes standard conversion routines found in most good monitors such as ASCII to hex, hex to ASCII, hex to decimal, decimal to hex, random number generators, console message output routines, console parameter input routines, etc. Using these routines when coding a program can often save a lot of time and work. For one thing, the routine is already debugged and working, and, for another, a simple CALL statement (or a JSR for 6800 machines) is all that is necessary to utilize the existing code. Thus the third part of

my proposal is:

1. Define and publish standard routines for permanent residence in all machines.

2. Utilize the routines specified via software vectors.

3. Add the necessary vectors immediately after the I/O vectors.

Should the second half of the proposal be acceptable, the start of every program might appear as in Program (B).

Because establishing any system of standards is a lengthy and difficult task, we have limited our proposal to three used most frequently standard conversion routines. Hopefully, by limiting the initial standardization attempt to documentation, I/O routines, and just three additional routines, these standards will enjoy a greater chance of being accepted and implemented. The proposed standard console message routine is coded in Program (C). To utilize this routine, the programmer simply loads the address of the byte containing the number of characters in the message into the H and L registers and, if necessary, saves the contents of the B and C registers on the stack:

```
LXI   H,MSG1
PUSH  B
CALL  CPMSG
POP   B
```

The data actually stored at the memory location labeled MSG1 is shown in Prog. (D) This coding technique gives the programmer complete freedom to output a message up to 256 characters long and to locate that message itself is short, which is important to those who wish to store their utility routines or their monitor in PROM or ROM.

The proposed ASCII-to-hex conversion routine is shown in Program (E).

1. Move the first, or high order, character to the B register and the second, or low order, character to the C register.

2. Call ATOH.

3. Upon return from ATOH the A register will contain the hex value represented by the two characters, if the carry bit is not set (carry = 0). If the carry bit is set to 1, one of the two characters did not fall within the range 0 through 9 (30H through 39H) or A through F (41H through 46H).

Routine NBL may be used for converting a single character by moving the character to the A register and calling NBL. Again, the carry bit will be set if the character is not 0 through F.

Routines which convert a binary value in the A register to two ASCII characters in the B and C registers are shown in (F).

To use the hex to ASCII routines, simply put the hex value to be converted into the A register, call HTOA, and retrieve the high order character from the C register and the low order character from the B register. You will note that the results of the conversion have been put in the B & C registers in seemingly reverse order. This was done to facilitate output which uses the C register as a communication device. An example: Program (G).

```
HTOA:   MOV  B,A        ;Save hex value
        RRC             ;Rotate to high order nibble
        RRC
        RRC
        RRC
        CALL CONV:      ;Convert high order nibble
        MOV  C,A
        MOV  A,B        ;Restore hex value
        CALL CONV
        MOV  B,A
        RET
CONV:   ANI  0FH        ;Clear upper 4 bits
        ADI  90H        ;Set for carry if A through F
        DAA
        ACI  40H        ;Add carry and adjust
        DAA
        RET
```

Program (F)

```
        CALL HTOA  ;With value in A register
        CALL CO    ;Output high order
        MOV C,B    ;Get low order
        CALL CO    ;Output low order
```

Program (G)

```
ORG  XXXX       ;Where XXXX is the "ORG" address plus 3
JMP  AAAA       ;Where AAAA is the address of your con-
                ;sole input routine
JMP  BBBB       ;Where BBBB is the address of your
                ;console output routine
JMP  CCCC       ;Etc.
```

Program (H)

Conclusion

For any system of standards to be effective it must be agreed upon by a large portion of all possible users and then adhered to. This is a lot to ask of the individuals currently involved in personal computing, and probably even more to ask of the various manufacturers marketing equipment to the hobbyist. But, should the implementation of these standards take place, the rewards for all will be well worth the effort.

If we in the personal computing community wish to put forth the time and effort, we might eventually place into widespread use an operating system monitor which includes utility routines, such as those described above, as well as others for all of the most widely used microprocessors (including 8080, 6800, 6502, Z80, 2650, F8, and others). Further, if the manufacturers care to cooperate, the monitors could be located at the same address in all machines, thereby eliminating the need for the software vectors described earlier.

To accomplish this, I suggest a standards conference perhaps similar to the Kansas City conference that produced the KC Cassette Standard, but much broader in scope.

In the meantime, for those who care to join me in implementing a minimum software coding standard for the 8080, here is a tip to easily utilize the I/O vectors and the utility routine vectors described earlier in the article.

Upon receipt of a piece of software conforming to these standards, ascertain the "ORG" (origin) address, and assemble the code in Program (H).

(For complete listing, refer to Listing under "External Routines.")

After loading the received program, simply load your vectors on top of it (or append your vector object code to the program object code). Save your vector source code so that, by merely changing the ORG value to fit subsequent programs, it need not be recoded, only reassembled.

I propose two final suggestions for this standard. The first, which applies to all machines, is to leave adequate room to add more vectors as required in the future. This can be easily accomplished by coding the first statement in the program (the JUMP instruction at the label BEGIN) as: JMP 100H. This will allow for a total of 63 I/O and utility routine vectors which should be more than enough.

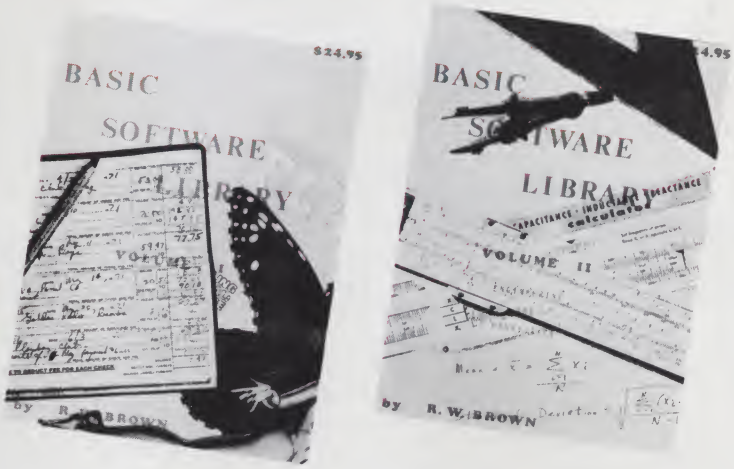
The second suggestion applies only to 8080 and Z80 machines, utilizing RAM addressed at location 0. Make the address of label "BEGIN"=40H, thus leaving free all 8 interrupt addresses for use by those whose systems employ interrupt driven I/O.

Other Machines

My prime area of interest is in systems and utility programming for the 8080 and Z80, thus I've yet to educate myself sufficiently in other machines to propose adequate utility routines. I would therefore like to invite those whose prime interest is in 6800, 6502, or other microprocessors to write up their suggestions and join me in the crusade for more useful software. ■

IF YOU LIKE COMPUTERS

YOU'LL LOVE OUR SOFTWARE



VOLUME ONE

Part 1

BOOKKEEPING
Bond
Building
Compound
Cyclic
Decision 1
Decision 2
Depreciation
Efficient
Flow
Installment
Interest
Investments
Mortgage
Optimize
Order
Pert Tree
Rate
Return 1
Return 2
Schedule 1

Part 2

GAMES
Animals Four
Astronaut
Bagel
Bio Cycle
Cannons
Checkers
Craps
Dogfight
Golf
Judy
Line Up
Pony
Roulette
Sky Diver
Tank
Teach Me

PICTURES
A. Newman
J.F.K.
Linus
Ms. Santa
Nixon
Noel Noel
Nude
Peace
Policeman
Santa's Sleigh
Snoopy
Virgin

VOLUME TWO

Part 3

MATH & ENGINEERING
Beam
Conv.

This BASIC SOFTWARE LIBRARY is a complete do it yourself kit. Written in everybodys BASIC immediately executable in ANY computer with at least 4K, no other peripherals needed. Over 700 pages of source codes, descriptions and instructions.

*** * Also available at most computer stores * ***

This Library is the most comprehensive work done of its kind to date. There are other software books on the market today but they are dedicated to computer games. The intention of this work is to allow the average individual the capability to easily perform useful and productive tasks with a computer. All of the programs contained within this Library have been thoroughly tested and executed on several systems. Included with each program is a description of the program, a list of potential users, instructions for execution and possible limitations that may arise when running it on various systems. Listed in the limitations section is the amount of memory that is required to store and execute the program in K Bytes.

Volume I contains Business and Recreational programs and is 300 pages long. Volume II is 260 pages long and contains Math, Engineering, Statistics and Plotting programs. Volume III contains money managing Advanced Business programs such as Billing, A/R, Inventory, Payroll, etc. All the volumes are printed on standard 8½" x 11" paper.

Volume I or II $24.95 each
Volume III $39.95 each

****** CHRISTMAS SPECIAL ******
Volume I & II $39.95 set
Volume I, II & III $69.95

add $2/Vol for handling; $4/Vol for air mail (Continental U.S.)

Money Orders and Bank Card orders shipped same day. C.O.D. and checks take longer. Personal checks may delay shipping up to 4 weeks. Pricing subject to change without notice. If air mail shipping desired add $4 per volume to price, Continental U.S. only. Foreign orders add $9 for each Vol.

Volume discounts available.

SCIENTIFIC RESEARCH

P.O. BOX 2096-K
ASHLAND, VIRGINIA 23005

Your BANKAMERICARD welcome here

master charge THE INTERBANK CARD

DEALER AND DISTRIBUTION INQUIRIES WELCOME.

(Vol.II cont.)

Filter
Fit
Integration 1
Integration 2
Intensity
Lola
Macro
Max. Min.
Navaid
Optical
Planet
PSD
Rand 1
Rand 2
Solve
Sphere Trian
Stars
Track
Triangle
Variable
Vector

Part 4

PLOTTING & STAT
Binomial
Chi-Sq.
Coeff
Confidence 1
Confidence 2
Correlations
Curve
Differences
Dual Plot
Exp-Distri
Least Squares
Paired
Plot
Plotpts
Polynomial Fit
Regression
Stat 1
Stat 2
T-Distribution
Unpaired
Variance 1
Variance 2
XY

APPENDIX A
BASIC STATEMENT DEF

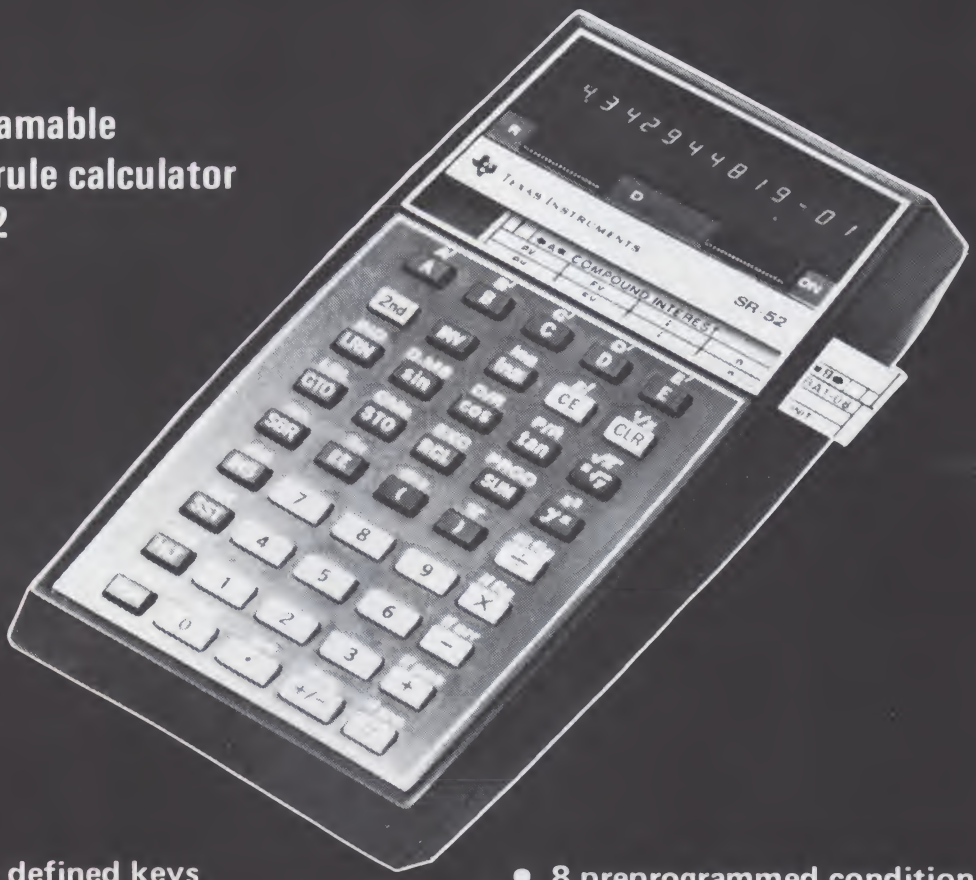
VOLUME THREE

Part 5

ADVANCED BUSINESS
Billing
Inventory
Payroll
Risk
Schedule 2
Shipping
Stocks
Switch

NOW FROM TEXAS INSTRUMENTS . . .
three machines in one.

programable
slide-rule calculator
SR-52



- 10 user defined keys
- 224 program storage locations
- 23 preprogrammed key functions
- 8 preprogrammed condition statements
- 20 independent addressable memory registers
- Permanent program storage on magnetic cards

It took TEXAS INSTRUMENTS to invent the SR-52 calculator. It took C & S MARKETING ASSOCIATES to offer it at a price you can afford, now only $249.95. With such versatility and such an affordable price, you can not afford to be without the problem solving power of card programability. Now solve problems in seconds that would take hours with any ordinary calculator or slideruler if they could be done at all.

For more information or the answer to any question you may have about the SR-52 calculator, call toll free (800-251-6771)*. Tenn. residents call (800-252-6706). Other TEXAS INSTRUMENT models available from $49.95.

Each TEXAS INSTRUMENT calculator comes with a 1-year warranty. Should your unit prove defective within 60 days, just return it for a new unit! Finally should you be dissatisfied with your calculator return it within 15 days for a prompt refund.

C & S MARKETING ASSOC.
P.O. BOX 165 ALGOOD, TENN. 38501

QTY. _____ PRICE 249.95 ea.

☐ CHECK ☐ M.O. ☐ C.O.D.

☐ LITERATURE ☐ INFORMATION

DIAL (800-251-6771) IN TENN. DIAL 800-262-6706

Practical Microcomputer Programing

... Part 1: Logical Instructions

John W. Molnar
Box 561
Ridgefield NJ 07657

This is the first in John Molnar's series on practical microcomputer programming. And, the use of the word "practical" is very appropriate because he has come up with some good examples to illustrate the use of logical instructions in the following article. (Be sure and catch his article on a proposed hobbyist operating system next month.) John is currently writing a book dealing with microcomputer programming. Should be a good one. — John.

Let's face it! The so-called "microprocessor revolution" is well under way. A random glance through any electronics magazine will reveal a wealth of articles and advertisements aimed at the home computer market. The microcomputer experimenter has a wide range of hardware to choose from ranging from simple "CPU prototype boards" all the way to elaborate systems complete with color graphics CRT display. Applications for micro based systems are also never ending; radio amateurs have developed systems to control complete radio teletype (RTTY) stations and antenna headings, while, at the other end of the spectrum, computer game freaks search for the perfect TREK program. And, of course, there are the classical computer applications, such as bookkeeping and inventory control. The key to the successful computer application rests only partially with the hardware system. Unfortunately, the major problem encountered by the computer experimenter is programming, that seemingly hopeless task of making the bits, bytes, registers, and instructions perform together to make an application work as well as it should. More often than not, the hardware assembly and debug of a micro system is the easiest part; the problems occur when the system is to be used. Compare a microprocessor kit, such as the SWTP 6800 and a Heath stereo amplifier. Once completed, the amplifier need only be plugged in and connected to other components to be enjoyed — the computer does nothing when completed until programmed by its user — a difficult task for the individual unfamiliar with the art of programming. To the uninitiated and inexperienced, programming is bewildering at best and downright frightening at worst.

It is the intent of this article, the first of a series, to lift some of the shroud of mystery concerning microprocessor programming. There are countless tips and techniques that, once understood, can make the software end of microprocessors as enjoyable as the building and testing of the kit.

It is assumed that the reader is familiar with the architecture of his or her particular micro, the registers available, the instruction set, and the data paths. Most micro systems today are programmed in assembly language, using the machine's actual machine code to develop the program. Although "high-level" language processors, such as BASIC, are available for some micros, techniques learned through assembly language programming can always be carried forward.

This article specifically covers use of the logical instructions AND, OR and EXCLUSIVE OR. Beginning programmers seem to misunderstand, and therefore avoid, these powerful instructions. This is unfortunate, as many programming shortcuts and algorithms are based on the logical instructions. Hopefully, this discussion will allow the programmer to bridge the gap between simply using the microprocessor to add numbers and the development of meaningful software. The examples presented are written in Motorola M6800 assembly language. The techniques, however, are definitely applicable to any micro instruction set that includes logical instructions. All the examples have been tested for accuracy on a Motorola 6800 system and the reader is invited to duplicate, modify and hopefully understand them on his or her own system.

Logical Instructions Allow "Bit Manipulation"

Computer instruction sets usually allow four types of operations: *arithmetic, input/output, control,* and *logical.* The first three classes are somewhat self-explanatory, especially after reading the programming manual packed with the micro kit. However, the logical instructions are often presented with a "truth-table" discussion involving a maze of ONE and ZERO bits that seemingly tell all — that is, all but what they

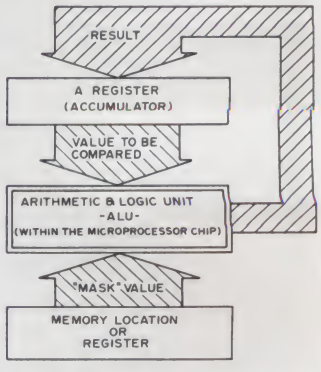


*Fig. 1. Logical Operations Block Diagram. (*Note: *the value to be compared in the Accumulator and the mask value are interchangeable.)*

do and how to do anything useful with them! The key to understanding logical instructions is to remember that they deal with *single* bits within a computer register or memory location as opposed to the arithmetic operations that produce a result based on the entire contents of a location. In an eight-bit micro, one logical instruction produces *eight* individual "answers," one for each bit in the computer word. We will show that the capability of manipulating single bits provided by the logical operations is the key to a multitude of programming techniques usable in every program.

The AND Function — or, It Takes Two to Get One

All logical functions work by comparing two bits in different locations to produce a single bit answer that replaces one of the original bits (see Fig. 1). The bits compared occupy the same position in each of the respective locations, bit position 0 in word 1 is compared to bit 0 in word 2, etc. The target location for the answer is usually a computer register. With this in mind, let's look at the first of three possible bit comparisons, the logical AND function. The rules for the AND are expressed in truth-table form in Fig. 2. Note that in order to produce a 1 result both of the compared bits must be 1s. A zero condition is produced if *either* or *both* of the compared bits are 0. It is most important to recall that *eight* unique answers are produced by a single AND instruction in an eight-bit micro. The state of any bit adjacent to the bits compared does not affect the result. Fig. 3 illustrates a simple program which loads the 6800 A register with the hexadecimal value 31, the ASCII representation of the printable character 1. (This example illustrates an important programming function allowed by the AND operation, that of *masking.*) Memory location 20 (hex) is initialized with the value OF. The AND A MASK instruction performs a logical AND between the contents of the A register and the contents of location 20 (designated MASK). The result, 01, is placed in the A register. If you don't have the capability of actually executing the sample program, lay the bits out on paper and perform an AND function, bit by bit. Your results should be the same.

BIT 1	BIT 2	RESULT
1	1	1
0	0	0
1	0	0
0	1	0

Fig. 2. AND function truth table.

It is often useful to isolate certain bits within a word while discarding (setting to zero) the remaining bits. An everyday programming example illustrates the point: The ASCII code, a means of representing data in a computer on external peripherals such as a TTY, is not usually suitable for internal computer operations. Table 1 shows the ASCII representation of the letters and numbers. Note that the ASCII value of the characters increases by one while progressing upward through the letters and numbers. This *collating sequence* is useful when we wish to compare ASCII data in computer programs, as the value for the letter B is less than that for C, etc. Also note that in order to represent the 256 possible ASCII characters it takes eight bits per character, many more than necessary if a program needs only to

```
LOCATION   CODE       INSTRUCTION   COMMENT
   00      86 31      LDA A,$31     A REGISTER = CHARACTER '1'
   02      C6 0F      LDA B,$0F     'MASK' VALUE TO B REGISTER
   04      F7 00 20   STA B,MASK    MASK TO LOCATION '20'
   07      B4 00 20   AND A,MASK    STRIP BITS 4-7
   0A      3F         SWI           STOP
    .
   20                 RMB 1         MASK LOCATION WHEN PGM EXECUTES
```

Fig. 3. Program using AND instruction to isolate bits 0-3 in the A Register.

CHARACTER	HEXADECIMAL VALUE	BINARY VALUE
0	30	0011 0000
1	31	0011 0001
2	32	0011 0010
3	33	0011 0011
.	.	.
.	.	.
9	39	0011 1001
A	41	0100 0001
B	42	0100 0010
C	43	0100 0011
.	.	.
.	.	.
O	4F	0100 1111
P	50	0101 0000
Q	52	0101 0010
.	.	.
.	.	.
Y	59	0101 1001
Z	5A	0101 1010

Table 1. Partial ASCII character table. Note that ascending character values have corresponding increasing numeric value. This "collating sequence" is useful when sorting characters in memory.

decode, or detect ten digits and 26 letters. The numbers require only four bits to make them unique, while the letters require six. It is often desirable to "strip" the unused bits from characters to make program calculations easier. *Masking,* using the AND instruction, simplifies the process of removing the undesired bits. It works like this: To save the desired portion of an ASCII byte, set the appropriate bits in the MASK to one while resetting (zeroing) the remaining bits. This mask byte should be in a memory location, not a register, so it will remain untouched by the AND operation that strips unneeded bits. Referring again to Fig. 3, we find the mask is the contents of location 20. Each time the A register is ANDed against the mask the most significant four bits are reset and the desired bits remain unchanged in the A register. A very practical example, Fig. 4, illustrates this technique. Let us visualize an amateur teletype operator who desires to control five station functions by using a micro system. The code letter A, B, C, D or E, is entered at the CRT to select the desired routine. In Fig. 4, it is assumed that a previous routine validated the input command character and left it in the A register in ASCII (eight bit) format. The routine required must somehow cause a branch to the desired computer routine RTN1-RTN5 based on the input character. Referring to Fig. 4, the selection routine works as follows: The A register is ANDed with a mask of 07 (hex) in location 25, resulting in the stripping of bits 3-7. Note that the

```
LOCATION   CODE          INSTRUCTION   COMMENT
   00      B6 00 24      LDA A,CHAR    CHARACTER A,B,C,D,OR E
   03      C6 07         LDA B,$07     MASK VALUE TO ...
   05      F7 00 25      STA B,MASK    MASK LOCATION 25
   08      B4 00 25      ANB A, MASK   AND, MASKING BITS 0-2
   0B      48            ASL A         SHIFT, DOUBLING VALUE IN A REG.
   0C      B7 00 23      STA A,OFSET+1 SAVE VALUE
   0F      FE 00 22      LDX OFSET     FETCH OFFSET VALUE
   12      EE 30         LDX TABLE,X   X=TABLE VALUE=ROUTINE ADDRESS
   14      6E 00         JMP X         EXIT TO SELECTED
   14      6E 00         JMP X         EXIT TO SELECTED ROUTINE
    .
   22      OFSET         FCB 0         INITIALLY ZERO
   23                    RMB 1         SAVE LOCATION FOR OFFSET VALUE
   24                    RMB 1         INPUT CHARACTER ("3" FOR EXAMPLE)
   25      MASK          RMB 1         MASK LOCATION
    .
   30      00 00 TABLE   FDB 0         DUMMY ADDRESS
   32      01 00         FDB RTN1      ROUTINE 1 ADDRESS
   34      02 00         FDB RTN2      ROUTINE 2 ADDRESS
   36      03 00         FDB RTN3      ROUTINE 3 ADDRESS
   38      04 00         FDB RTN4      ROUTINE 4 ADDRESS
   3A      05 00         FDB RTN5      ROUTINE 5 ADDRESS
```

Fig. 4. Routine Selection Program discussed in the text. One of five routines, RTN1-RTN5, is selected based upon input characters A-E. The input character is deposited in location 24 by an input routine (not shown). The RMB directives cause the assembler to reserve memory bytes for use by executing programs. The FDB, or Form Double Byte directive, causes the assembler to form actual data for the program to use. In this case, that data is the address of a routine elsewhere in memory called RTN1-RTN5.

```
LOCATION   CODE       INSTRUCTION   COMMENT
   00      CE 01 00   LDX #100      FIRST LOCATION TO CHECK
   03      C6 40      LDA B,#40     MASK VALUE TO ISOLATE BIT 6
   05      17  LOOP   TBA           MASK TO A FOR CHECK EACH CYCLE
   06      A4 00      AND A,X       SEARCH FOR BIT
   08      26 07      BNE FOUND     HIT, ENTER FOUND ROUTINE!
   0A      08         INX           INCREMENT X FOR NEXT CYCLES CHECK
   0B      8C 01 80   CPX #180      DONE??
   0E      26 F5      BNE LOOP      NO, CONTINE CHECKING
   10      3F         SWI           YES, HALT
           FOUND      ...           PROCESSING ROUTINE FOR HIT
```

Fig. 5. Routine for memory search featuring the AND instruction. When memory byte with bit 6 is found, the routine exits for additional processing at point FOUND.

ASCII representation of the characters A-E in Table 1 make the resulting A register value a binary number in the range 001 for A to 005 for E. This binary value is the key to our selection process. Since we want to form the *address* of the desired routine and since all addresses in the 6800 micro are 16 bits in length, the eight bit value in the A register must be expanded to a 16-bit address. This is accomplished by *shifting* the A register one bit to the left, effectively doubling the numerical value of the A register and producing an "address compatible" value. (*Note:* Using the SHIFT function of most micros is an easy way to multiply or divide a value by any power of two. The left shift multiplies the value by a power of two for each shift; a right shift divides the value.) Assuming that the operator entered the character "C" to select the third of five routines, we find the binary value in the A register to be 00000011 after the AND operation and 00000110 after the shift. The routine is now ready to actually form the address of the subroutine desired. The A register is stored in location 23, location 22 having been previously cleared. The instruction LDX OFFST then loads the values at locations 22 and 23 into the 6800 index (X) register, which is 16 bits wide. The hexadecimal value in the X register is now 0006. A second X register operation LD TABLE,X loads the X register with the address of the desired routine. Location TABLE and the locations following contain the five routine addresses. The value in the X register is formed by the internal machine operation that adds the existing value in the X register (0006) to the address of TABLE. Thus, the actual value in X is TABLE+6, which is the address of the third routine, the one desired. All that we now need is a jump to the routine using the X register for the jump address. Thus, a simple program has allowed the transition between an input character and the routine it represents. This example illustrates a simple, yet powerful, means of directing computer operations based upon outside-world human inputs. The only restriction on this program is that the TABLE of addresses must be within the first 256 memory addresses as the offset value for a 6800 index operation is 256 bytes. Note that had the AND instruction been eliminated the offset value before the shift would have been 67 instead of 3 (the binary value of the character C). Step through the example, eliminating the AND instruction, and see for yourself.

BIT 1	BIT 2	RESULT
1	1	1
0	1	1
1	0	1
0	0	0

Fig. 6. Truth table for the OR function

A final practical example of the AND instruction is shown in Fig. 5. In this example, we want to scan all memory locations between 100 and 17F (hex) for any byte with a value of 64. Knowing that 64 is equal to 2^6, we must check bit six of every location for a 1 condition. The program functions by using the AND to isolate the desired bit. Each cycle through the loop causes successive locations, starting at 100, to be checked for bit six being set. If this is found, control is passed to the processing routine FOUND. Note that the A register is reloaded with the mask each cycle, since, in this case, the register, not the memory location, is altered as the programs executes. Loop control is provided by insuring that the X register has not exceeded the final location (17F) each time it is incremented by the INX instruction. When the X register does exceed 17F, the program halts.

In summary, there are two important uses of the AND instruction to be remembered: The first is to clear undesired bits from a location using an appropriate mask; the second is to check for the existence of a SET bit by using a mask with the desired bit set.

```
LOCATION   CODE       INSTRUCTION   COMMENT
   00      BA 00 20   ORA A,$20     "OR" FIRST PATTERN INTO A REG
   03      BA 00 23   ORA A,$23     "OR" SECOND PATTERN
   06      BA 00 25   ORA A,$25     AND LAST PATTERN, PRODUCING "CD"
   09      3F         SWI           HALT
   .
   .
   20      41         FDB $41       FIRST PATTERN
   23      80         FCB $80       SECOND PATTERN
   25      0C         FCB $0C       THIRD PATTERN
```

Fig. 7. Example of logical OR function used to combine bit patterns into single register word. The final pattern is CD (Hex) or 11001101.

The Logical OR — Any 1 is a 1

Observing the truth table of Fig. 6, it can be seen that the OR operation is the opposite of the previously discussed AND. When *either* or *both* of the bits compared are 1, the result is 1. The OR function also operates on a bit by bit fashion within the computer word. However, this time the mask is used to SET any bit in the word. This is the primary use of the OR function, which is used to *connect* or combine isolated bit patterns into one word. Fig. 7 illustrates the OR in action. Locations 20, 23 and 25 contain different bit patterns; respectively 01000001, 10000000, and 00001100. The three ORA A instructions combine the pattern in the A register, each pattern places its 1 bits into the respective bit position in the register. The final result of the operation is 11001101. The capability of the OR to set any bit has useful programming ramifications. Returning to the ASCII character example of Fig. 4, we recall that the AND was used to CLEAR undesired bits from the character for internal processing. However, in order to output an internal binary value, we must restore the correct bits to form a valid ASCII character. Assume that the binary value 1001 (9 in decimal) is contained in the lower bits of the B register. It is desired to output this value to the teletype, but the bit pattern 00001001 is not the correct ASCII code for 9 (the correct value is 00111001). Hopefully, it will now be obvious that in order to convert the unprintable version of 9 to a printable form the two missing bits must be added to the word. The OR function may be used to mask (or add) the bits into the word, as shown in Fig. 8. A mask of 00110000 is set up in loca-

```
LOCATION   CODE       INSTRUCTION   COMMENT
   00      F6 00 30   LDA B,VALUE   B REG = 1001, A NON-PRINTABLE VALUE
   03      FA 00 20   ORA B,MASK    MASK IN VALUE 00110000
   06      7E 01 00   JMP PRINT     MAKING THE RESULT PRINTABLE.
   .
   .
   20      30  MASK   FCB $30       THE MASK
   .
   .
   30      09  VALUE  FCB $09       INITIAL NON-PRINTING VALUE
```

Fig. 8. Example of logical OR function used to mask (combine) necessary bits into a binary value representing a printable ASCII character.

```
LOCATION   CODE      INSTRUCTION     COMMENT
  0100     B6 01 20  LDA A,FLAG      FETCH FLAG WORD
  0103     BA 01 21  ORA A,FIGS      SET DESIRED FLAG BIT
  0106     B7 01 20  STA A,FLAG      RESTORE FLAG WORD
    .
    .
  0120        FLAG   FCB 0           FLAG WORD
  0121     08 FIGS   FCB $08         MASK TO SET "FIGS" FLAG
```

Fig. 9. Using logical OR to set a flag bit for program control.

tion 20. This is ORed with the B register, producing the printable ASCII character 9. Note that the OR does not disturb existing bit patterns when the mask bits are zero. This feature suggests another OR use, that of maintaining program *flags*. Many times a program, when executing, requires the capability of remembering whether events or steps have or have not been performed. For example, in the radio teletype program mentioned earlier, the program may need to know the mode of the RTTY terminal. If the terminal is in the FIGS (figures) mode, numbers and special characters are printed; when it's in the LETS (letter) mode, letters are printed. A single bit, or *flag*, may be used by the program to determine if the carriage requires a shift before outputting a data character. A word called FLAGS exists in the program with bit three of the word indicating the TTY mode. If bit three is set, the carriage is in the FIGS mode, and, if reset, the LETS mode. Assume that the program desires to send a number to the TTY, requiring the carriage to be in the FIGS position. Before sending a FIGS character, the program sets the bit three flag using an OR instruction with a mask word of 00001000. Now, when a later program module desires to send a letter, the FIGS flag can be checked, and, if set, a LETS character will be sent to the TTY before the data is transmitted. (*Question for thought:* How did the output program check the state of the FIGS flag? *Hint:* Possibly a previously discussed logical function was used!) See Fig. 9 for practical program details. In concluding the discussion of this example and the OR instruction, note that the other seven bits in the FLAG word are not affected when one is set, as long as the corresponding mask bits are 0. The technique of using flags to control program sequencing is important, as it great simplifies the intercommunication links between program modules.

And Finally, The EXCLUSIVE OR — or How to Flip a Bit

We have just shown that the OR function may be used to selectively set a bit, a useful technique for setting program flags. However, in order for this technique to be effective, a method for *resetting* the flag bit on request must exist. The last of the three logical functions, the EXCLUSIVE OR (XOR) is used to that end. The truth table for the EXCLUSIVE OR is represented in Fig. 10. The result of a bit comparison is a 1 only when the bits checked are of *different states*. Either two 0s or two 1s result in a 0. Thus you can see that the XOR function can be used to alternately SET and RESET a bit in a register or memory. For example, if the A register contains the value 01001000 and the XOR mask is 11111111, the EOR A instruction results in the value 10110111 being placed in the A register. If the reversed A register is again XORed against the same mask, the original pattern is reproduced, *ad infinitum*. (Refer to Fig. 11.) The XOR function may be used to reset flag bits previously set. Recalling the example of Fig. 9, the FIGS flag bit can be reset by using the same mask used by the OR instruction that originally set the bit. Again, note that the only bit altered is the one reflected by a set mask bit. All the others remain unaffected under the rules of the XOR. *Note:* The XOR with mask could also have been used to set the bit originally as long as the flag word was initially zeroed (which is a good practice, anyway). Check it out.

BIT 1	BIT 2	RESULT
1	1	0
0	1	1
1	0	1
0	0	0

Fig. 10. Truth table for EXCLUSIVE OR (XOR) function. Unlike bits produce a ONE result; like combinations a ZERO.

```
LOCATION   CODE      INSTRUCTION     COMMENT
  00       86 48     LDA A,INIT      INITIAL PATTERN = 01001000
  02       B8 00 20  EOR A,MASK      "FLIP" PATTERN TO 10110111
  05       B8 00 20  EOR A,MASK      RESTORE INITIAL PATTERN
  08       3F        SWI             HALT
    .
    .
  20       FF MASK   FCB $FF         MASK
             INIT    EQU $48         INITIAL PATTERN
```

Fig. 11. Program example in which the EXCLUSIVE OR instruction is used to alternately flip the state of a bit.

In Conclusion

So, that's it for the logicals! The AND, OR and EXCLUSIVE OR deal specifically with *bits* in a computer word, and . . . knowing how they work, the reader should be able to put them to good use in individual application programs. Liberal use of the logicals will result in shorter, more efficient programs and in many cases save considerable debug time when a previously written area of code can be eliminated. In future articles the use of additional programming techniques will be covered, including: the *condition code*, signed numbers, sorting methods, and routines dedicated to testing a new microprocessor and its memory after assembly. Familiarity with these simple, confidence building routines will help you develop and exercise new hardware and software skills. ■

NEW PRODUCT or SERVICE ?

The Kilobaud list of DEALERS, CLUBS, PUBLICATIONS and MANUFACTURERS is by far the most complete available (it's one we use for our mailings and we update it daily). The list has over 600 names painstakingly gathered from manufacturers, magazine ads and new product releases, hobby computer shows and direct mail.

We'll print this list out on self-sticking labels for you for $50. Additional printouts, once you're a customer, are $35. If you're in the industry (rated) just call in your order — otherwise call in your order or send it with check or charge information (BAC, AMEX, MC). Our toll free number for these orders is 800-258-5473.

NEW FIRMS, DEALERS, CLUBS . . . be sure we have your name, address, phone number and as much data as possible for this listing.

kilobaud PETERBOROUGH NH 03458 kilobaud

Dr. Lance Leventhal
Emulative Systems Co.
138 So. Acacia Ave
Solana Beach CA 92075

Well, Your Micro's Built

... where do you grow from here?

Lance Leventhal teaches microcomputer systems design (among other things) and seems to have a real knack for getting down to the grass roots level when he writes about the little beasts. His article discusses microcomputer architecture (particularly the memory and I/O sections) and provides us with some good introductory material on the subject (an introduction to microprocessor architecture by a PhD . . . but not for PhDs). — John.

Many novices to the computer field see a computer kit advertised for a few hundred dollars and think the kit will be all they need for whatever project they might imagine. However, they soon discover they can do almost nothing with the basic kit. They have no convenient way to enter data into the computer or get results from it, no way to attach peripherals or equipment, and no place to store data or programs. Of course, the company that supplied the kit will provide all the additional circuit boards the user needs, but the user soon finds the cost of the computer kit was a small part of the cost of the total system. A device like the IMSAI 8080 (shown in Photo 1) may look like everyone's idea of a computer, but it can't do very much without additional internal circuitry and external equipment.

But, you may ask, "I thought the microprocessor only cost $10 or $15 and that it was a complete computer-on-a-chip." Why does a working system cost thousands of dollars? The answer is that the microprocessor is not a complete computer, but only a section of one. Although the cost of the microprocessor is small, the rest of the computer is still expensive (although it, too, is becoming cheaper). The situation is as if automobile engines had become free; automobiles would still be quite expensive because of the body, seats, controls, transmission, other equipment, and labor. A manufacturer of a microprocessor-based product remarked at a recent conference that "The single most expensive item in our product



A Small Computer, the IMSAI 8080 (Courtesy of IMS Associates Inc.).

is the case. If we didn't make the case ourselves, we really couldn't make any profit."

Newcomers to computers often don't understand that a computer is not just a single enormous device. In fact, a computer consists of several distinct sections which can vary greatly in complexity. Not only does the speed vary but so does the size of their memories, the number and speed of their input/output channels, the types of tasks which they are best suited to perform, and the number and type of devices (called *peripherals*) and programs (called *software*) which are available with them. A total computer system is more than just an electronic brain, the brain must have a memory to store information, channels over which to transfer information to and from the outside world, and input/output devices which prepare data or record results. A computer without these features is like a person who can think but cannot remember and has no senses to provide data or muscles which the brain can direct.

Computer Structure

A computer consists of three basic sections: the control section (often called the central processing unit or CPU), the memory section, and the input-output section. Fig. 1 shows the sections and the paths which connect them. The control section actually processes the data; it gets the instructions from memory, decodes and executes them, performs arithmetic and logical functions, transfers data to or from the memory and input/output devices, and provides timing and control signals. The memory section contains storage for instructions and data. The input/output section handles the communications between the computer and the outside world. Signals of various types flow from one section to another along paths called *buses*. A microprocessor consists only of a CPU. Memory and input/output circuitry must be added to make a working computer or *microcomputer*. A simple, cheap microcomputer like the one shown in Photo 2 is possible only if the memory and input/output requirements are quite limited. This article will discuss the memory and input/output sections of computers; the next article in the series will discuss the CPU.

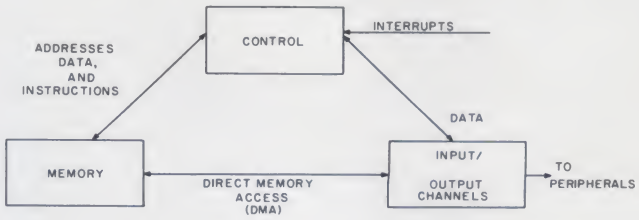


Fig. 1. Block Diagram of a Typical Computer.

Memory Section

The memory section is the simplest section to describe. The memory consists of cells which have two stable states. We call one of these states "zero" and the other "one"; thus we can use each cell to represent a binary digit or *bit*. Since we usually want the CPU to handle more than one bit at a time, we organize the binary cells into groupings of fixed length. The smallest grouping is a *byte* which generally contains 8 bits. The basic grouping of bits which the control section can handle at one time is a *word*. Computer word lengths vary from 4 to 64 bits and are generally the bit lengths of the control section's internal structure and the computer's data paths. Note that, in the case of an 8 bit CPU like the Intel 8080 or Motorola 6800, a word and a byte have the same length.

Word length is one measure of the power of a computer. A computer with a longer word can do more work than one with a shorter word, even if both computers operate at the same speed. The word length determines the amount of work done during each cycle. Large computers have long words — 32 bits (IBM 360), 36 bits (Univac 1110), or even 64 bits (Control Data 6600). Minicomputers have shorter words — 12 bits (Digital PDP-8) or 16 bits (Digital PDP-11 or Data General Nova). Microcomputers have still shorter words — usually 8 bits (Intel 8080, Motorola 6800, and MOS Technology 6502).

Obviously, in order to use a particular grouping of bits in a memory, the CPU must have some way of identifying that grouping. The identification used to select a particular byte or word is called an *address*. The address identifies the position of a byte or word, not its contents. Remember the difference between data and addresses! For example, byte #25 can contain any 8-bit number, not just the number 25. We

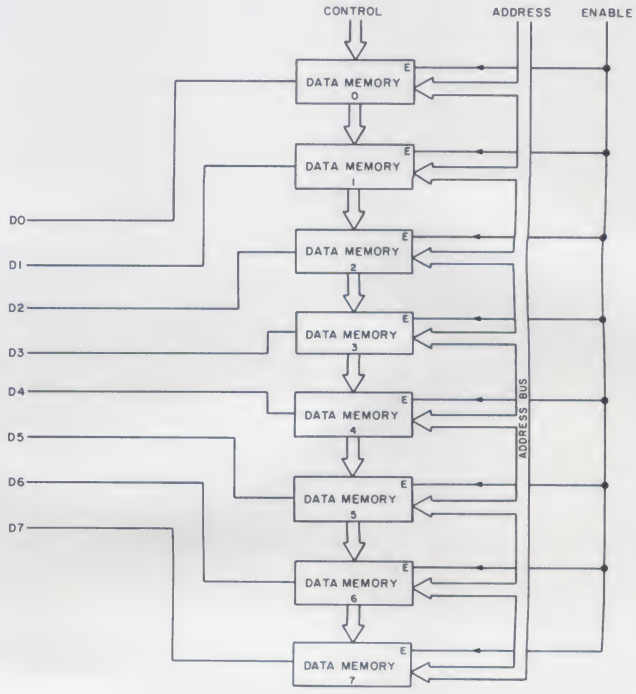


Fig. 2. Memories with single bit words can be combined in parallel to form the longer words needed for computer memories. All the 1 bit memories have the same CONTROL, ADDRESS, and ENABLE connections. The only difference is that the data line on each memory is connected to a different line of the computer's data bus.

Address Length (Bits)	Memory Size (Bytes)
8	256
9	512
10	1K
11	2K
12	4K
13	8K
14	16K
15	32K
16	64K

Table 1.
Memory Size vs. Address Length.

often use parentheses to indicate "contents of" and avoid this confusion. Thus 1000 means address #1000 and (1000) means the data which is in address #1000.

Each byte or word of memory needs a separate address. The bit length of the addresses determines how many words of memory can be easily attached to a CPU. Each time we add a bit to the length of the address, we double the memory capacity. Table 1 shows the relationship between the bit length of the addresses and the memory capacity for addresses between 8 and 16 bits long. Most small computers use 12 or 16 bit addresses so that they can handle 4K or 64K bytes of memory (1K = 1024 or approximately 1000). Note that the length of the addresses and the data is often not the same. Most microprocessors use 8 bit data words and 16 bit addresses. This allows the user to attach a reasonable amount of memory to the processor, but causes difficulties in handling addresses.

Both data and instructions are stored in the memory section of most computers. A computer which has data and instructions in the same form is called a *Von Neumann machine* after the mathematician who first proposed this method of storing programs. How do we tell whether a number is data or an instruction? The answer is that we can't tell. During an *instruction fetch cycle,* the CPU takes whatever number it gets from the memory and places the number in the instruction register. During a *data fetch cycle*, the CPU places the number it gets in a data register. For instance, in the case of the Intel 8080 CPU, the binary number 00001001 would be the instruction DAD B (double add with register B) if the CPU obtained it during an instruction fetch cycle and the number 9 if the CPU obtained it during a data fetch cycle. But couldn't the CPU get confused and start executing the data? The answer is yes, if we don't direct the CPU to the right place in the memory. The programmer must ensure that the computer gets data and instructions in the correct order. The computer will do whatever it is told to do, whether it makes any sense or not.

The memory section contains other devices besides the memory itself. The address decoder uses the addresses to select particular memory locations. Other circuitry may provide control signals for the memories, allow the memory section and control section to communicate properly, and ensure that the memory section retains its contents. We use standard integrated circuits or special controllers to preserve the contents of memories. We use other integrated circuits in the memory section as decoders, buffers, and latches. Thus the memory section of a small computer usually consists of several circuit boards, each containing a few thousand words of memory and some supporting circuitry. A small computer has room for these memory boards, each of which will typically cost one to two hundred dollars.

Semiconductor Memories

Virtually all hobby computers use semiconductor memory chips to store programs and data. These chips provide large amounts of storage in a compact form at very low cost. Larger chips and other associated LSI devices promise to make semiconductor memories even cheaper in the future.

Three major types of semiconductor memories exist: *ROMs* (read-only memories), *PROMs* (programmable read-only memories), and *RAMs* (random-access read-write memories). ROMs have their contents determined at the factory as part of the manufacturing process. A rather large masking charge (usually several hundred dollars) is involved; so ROMs are only economical when a large quantity with the same pattern is ordered. Thus most hobbyists will only use special-purpose ROMs which can be purchased without a masking charge. These ROMs include character generators for CRT displays and printers, various code converters (such as Selectric to ASCII), and tables such as sines or cosines.

Hobbyists are far more likely to use the PROM, a read-only memory which can be programmed by the user under special conditions. PROMs can have their contents changed by means of specified high voltage pulses. A PROM has most of the advantages of a ROM — it is *non-volatile* so it does not lose its contents when the power is turned off, it cannot be accidentally changed by an erroneous user program or computer malfunction, and it can provide a rather large amount of storage on a single chip at low cost. PROMs are somewhat more expensive than ROMs but can be purchased in small quantities without a masking charge. Note that the PROM cannot be altered during ordinary computer operation.

PROMs are available in many semiconductor technologies; but hobbyists usually find the erasable MOS PROMs the cheapest and easiest to obtain. We often call such PROMs *EPROMs* or *EROMs*. The most popular PROM is the Intel 1702A which has 256 8-bit words. The 1702A is relatively slow (1 or 1.5 microsecond access time), but is cheap ($5 to $15) and available from many sources. Larger PROMs (512 or 1024 8-bit words) and faster PROMs (access time around 500 nanoseconds) are available but cost much more than 1702As. The price of PROMs is continually dropping however; the larger and faster PROMs which now cost $30 to $50 each will probably be in the $20 price range by next year.

Hobbyists may choose to purchase, program, and erase their own PROMs. PROM programming circuits can be constructed or purchased in kit form. However, these circuits require high voltages which often must be maintained to within a rather close tolerance, so most hobbyists will prefer to let PROM or hobby computer vendors program or erase and reprogram their PROMs for a small extra charge. Most erasable PROMs (like the 1702A) must be placed under an ultraviolet light source for about ten minutes to erase the contents. The entire PROM is erased; single bits or words cannot be corrected. Some new PROMs like the AMI S6834 (a 512 x 8 PROM designed for AMI/Motorola 6800 systems) are more flexible; single locations can be erased by means of a single high voltage input without

even removing the PROM from the circuit board.

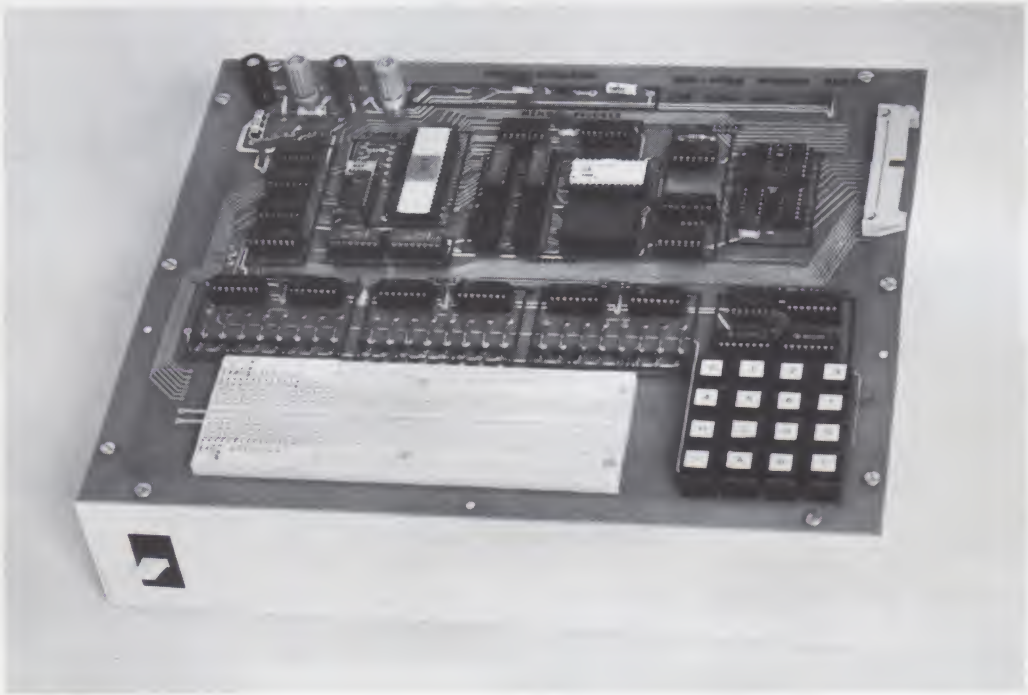
Many hobbyists will prefer to purchase preprogrammed PROMs. The ones now available generally contain debugging packages and system monitors. However, sophisticated loaders, assemblers, compilers, and other programs should be available in PROMs in the next few years. Larger PROMs at lower prices will make such PROM-based software economical. In fact, Microcomputer Technique now offers a PROM-based Intel 8080 assembler, and Microcomputer Associates offers a PROM-based MOS Technology 6502 assembler. PROM-based software is convenient since it doesn't have to be loaded into memory each time it is used, and it can't be written into or accidentally changed. The PROM may simply be removed from the board or system if the memory space is needed for other purposes.

Although PROMs are useful for software packages and for final versions of programs, the basic memory in most hobby computers will be RAM. Random-access really means that all locations can be reached in the same amount of time; in fact, ROMs and PROMs are random-access, although the term is usually reserved for read/write memories — shift registers and cassette tapes are examples of memories that are not random-access. RAMs are significantly more complex than ROMs or PROMs because of the need to both read and write data at the same voltage levels. RAM memory chips therefore contain fewer bits (4K bits is a large RAM, while PROMs are available commonly in sizes up to 8K and ROMs up to 16K). However, single-chip RAMs with more bits are becoming available; in the last ten years, the largest single-chip RAM has increased in size from 256 bits to 16k bits while its cost has remained about the same.

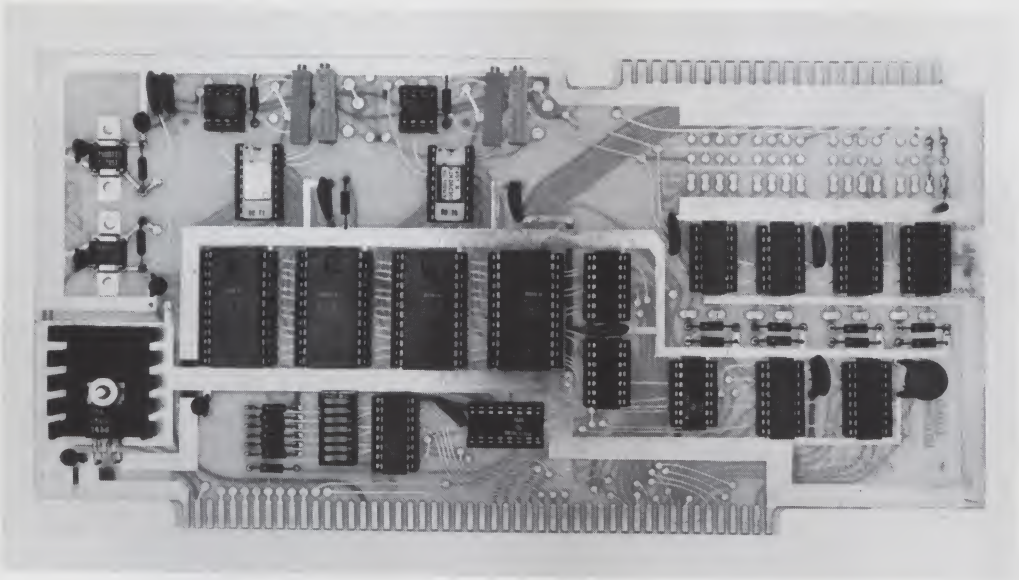
The standard RAM at the present time is the 1K by 1 2102-type device. Currently such RAMs sell for about $1 to $2 each. Note that each 2102 only has a 1-bit word; in order to form 1K 8-bit words, 8 2102s must be connected in parallel, with each RAM connected to a different bit of the data bus. (See Fig. 2 for an illustration of how we wire the memories.) Although the resulting word may be considered (and addressed) as a unit, each bit of the word actually resides in a separate 2102 memory. Several versions of the 2102 are produced which vary in access time and power consumption. A memory with an access time of about 500 nanoseconds is required to run an Intel 8080 or Motorola 6800 processor at full speed.

Larger RAMs are available at higher cost. 4K RAMs are often used in the memory boards which can be purchased from many manufacturers; they cost $20 in small quantities but much less in large quantities. Like the 2102 1K RAMs, most 4K RAMs are organized into 1 bit words; so they must be connected in parallel in order to be used as computer memories. MOst 4K RAMs are *dynamic*, i.e., they lose their contents unless that information is written back into them every once in a while. Such RAMs use capacitive memory cells which discharge after a certain amount of time. A typical dynamic RAM must have its contents restored (or *refreshed*) every 2 milliseconds; memory boards will contain the circuitry to perform the refresh, which is accomplished automatically so that the computer user is unaware of it (although it does slow the computer somewhat). Dynamic RAMs are cheaper, denser, and use less power than the so-called *static* RAMs which do not have to be refreshed. New LSI chips are becoming available to handle refresh; some processors (such as the new Zilog device) even include on-chip refresh circuitry. Future developments in RAMs will include faster devices (MOS RAMs are now available with access times of 100 nanoseconds), larger devices (16K single-chip RAMs are now being tested), and devices with more convenient organizations (such as 8 or even 16 bit words).

All RAMs currently are *volatile*, i.e., they lose their contents when power is removed. Thus programs or data have to be reloaded each time power is accidentally lost or the computer is turned off. Obviously, volatility is an inconvenient feature when a long program must be reentered from a slow paper-tape reader or keyboard. A small battery can be used to provide backup power if needed. CMOS memories



A Simple Microcomputer, the E and L Instruments Mini-Micro Designer (Courtesy of E and L Instruments Inc.).



An Analog I/O Board (Courtesy of Polymorphic Systems).

often have a low-power standby mode whereby data can be retained with a very small power drain. However, such memories are rather expensive and are only available in small sizes. While the development of a nonvolatile RAM is several years away, low-power RAMs which can operate with small batteries (or smaller power supplies) will become more widely available in the near future.

Semiconductor memory chips are a key part of hobby computers. PROMs can be used for final versions of programs and for monitors and other standard software. RAMs are the basic for almost all user memory. Lower prices, larger sizes, more convenient organizations, and i.e., 1, 2, 4, or 8 bit, and lower power consumption will serve to make semiconductor memory chips even more useful in the future.

Input/Output Section

The input/output section of a computer is an interface between the computer and the outside world. The computer is a digital electronic device which has its own internal clock and uses certain specified voltage levels to indicate binary zeroes or ones. Obviously few external devices operate in the same way as the computer. The *input/output* section must convert the input data sent from the outside world to a form the computer can understand and convert the output data sent by the computer to a form the outside world can understand. This conversion may be a very complex task. Often the CPU will perform part of the conversion; some large computers have separate processors which handle input/output tasks.

As with memory storage, the CPU handles input and output data in binary form. The CPU may handle a single bit at a time (*serial* input or output) or it may handle several bits at once (*parallel* input or output). Each separate grouping of input or output lines is called a *port*, which may have any bit length (although ports with the computer's word length are often the most convenient). Each port must have an identifying address so that the computer can direct data to or from it. The addresses may be determined by the same decoding system used for the memory section; this method is called *memory-mapped input/output* and means that the computer handles memory locations and input/output ports differently.

The input/output section will have to consider many characteristics of the transmitted signals in order to convert the signals into their proper input or output form. Among the characteristics which must be considered are:

1) Signal level

External devices, even when using two voltage levels to represent the two possible values of a bit, may use different voltage levels than the computer. Most computers use TTL (Transistor-Transistor Logic) levels, i.e., 0 volts for a binary 0 and +5 volts for a binary 1. However, some terminals use 12 volt signals, and gas ionization displays require signals of 150 volts or more in order to ionize the gas and provide visible light. The input/output section must convert signals from one set of levels to another.

2) Signal type

Many external devices don't use binary voltage levels. They may use current levels like a standard teletypewriter. They may use continuous (analog) signals of various types like a motor or thermometer. The input/output section must convert input data into the voltage levels which the computer understands and convert output data into the types of signals expected by external devices. The analog I/O board from Polymorphic Systems (shown in photo) contains latches (Intel 8212), decoding circuitry, and the converters needed to interface between the digital computer and the analog world.

3) Signal duration

A computer works much more rapidly than most external devices or observers. It will send its output data only as a very brief pulse. The input/output section must hold or *latch* the computer's signals so that an output device will have time to respond to them. For example, we won't be able to see a light unless it is lit for a tenth of a second; however, unless the input/output section holds the output data, the computer will turn the light on only for a few microseconds. The input/output section also must latch input data so that the computer will be sure to see it even if the computer is performing other tasks at the time the data arrives.

4) Signal timing

The computer accepts inputs and produces outputs at times determined by its own clock. External devices, however, work at their own pace in a way that rarely coincides with the computer



The Teletype Model 33 ASR Data Terminal, a Standard Teletypewriter (Courtesy of Teletype Corporation).

clock. Thus the input/output section must reconcile the timing of the computer and the external devices. It must hold the signal from external devices and send them to the computer when the computer requests them; it must also hold signals from the computer and send them to the external devices at the proper time.

The computer and external clocks are often very different. A typical computer, for instance, can transfer 100,000 characters per second while a typical teletypewriter (see photo) can transfer just 10 characters per second. Note the tremendous range of speeds involved. A computer may have to handle data from a temperature sensor which provides a reading every five minutes, a teletypewriter which operates at 110 bits per second, and a floppy disk (see photo) which transfers 250,000 bits per second.

5) Signal format

External devices often produce or require data in special formats. The external devices may expect data serially (i.e., one bit at a time). They may require or use special signals to indicate the start or end of a message. They may use special codes which reduce the probability of transmission errors. The input/output section will have to convert data between the format used by the computer and the format used by the input or output devices.

Clearly the input/output section of a computer can be quite complex. In simple cases, standard TTL latches hold input or output data; buffers and drivers ensure that the computer transfers signals correctly. Level translators provide the correct voltage levels, while analog to digital (A to D) and digital to analog (D to A) converters interface between analog devices and the digital computer. We often use special integrated circuits to format signals properly and resolve timing differences. A universal asynchronous receiver/transmitter (UART) performs serial/parallel conversions, provides proper formats and timing, and checks for various types of errors. Other devices that are often used include serial and parallel interfaces which we can program from the CPU to perform a variety of system functions. Thus the input/output section of a



A Floppy Disk System (Courtesy of IMS Associates Inc.).

small computer will consist of several circuit boards, each containing devices specially designed for specific input/output functions. The input/output section will also contain the decoding mechanism required to identify particular ports from their addresses.

A single I/O board can involve a variety of circuits. The Polymorphic Systems Video Interface Board (shown in photo) contains decoders, latches, buffers, counters, timing circuitry, keyboard connections, random-access memory to hold the characters displayed on the television screen, a read-only memory to generate the character patterns, and other devices required by the keyboard and video display. A typical backplane (or motherboard), such as in the Altair System shown in Photo 7, will have room for several I/O boards, each of which will cost one to two hundred dollars. The CPU, which was once the most expensive part of the computer, is now the cheapest. The new challenge for LSI technology is to greatly reduce the cost of the memory and input/output sections of microcomputers.

Connecting The Sections

Obviously we must connect the various sections of the computer. *Data buses* transfer data to or from the memory or input/output sections. *Address buses* transfer addresses to the memory or input/output sections. *Control buses* transfer control information between the various sections. In small computers, one bus often serves several purposes, i.e., the bus is time shared or *multiplexed*. Then the CPU must use control signals to indicate what is on the bus at a particular time.

Two special types of connections deserve further comment. One is the *interrupt*, a signal that demands the immediate attention of the CPU. Interrupts inform the CPU of the occurrence of external events such as an alarm condition or a peripheral which is ready to send or receive data. The CPU does not have to wait for the external events or check the state of external devices. Instead, the CPU can perform other tasks until it is interrupted and then resume those other tasks after servicing the interrupt. Interrupts are useful for inputs which occur irregularly at rates much slower than the normal operating speed of the computer.

Direct memory access is a method of transferring data between the memory and input/output sections of the computer without involving the processor at all. The CPU does not have to fetch and store the data or fetch and decode the instructions which would be required to transfer the data. Direct memory access (DMA) can thus handle data much faster and more directly than the normal input/output channels. DMA systems are used for devices like disks or CRT displays requiring a data rate comparable to or faster than the operating speed of the computer.

Interrupts and DMA are very convenient for handling different types of input/output. However, both methods involve special circuitry which further increases the cost of the computer system. As with many other items, the price of the basic computer may be lost in the price of that extra equipment which makes the computer system most useful and convenient. ■

References

Osborne, A., *An Introduction to Microcomputers* – Volume 1: Basic Concepts, Adam Osborne and Associates, Berkeley CA., 1976 Chapters 3 and 5.

Hilburn, J. L. and Julich, P. N., *Microcomputers/Microprocessors*, Prentice-Hall, Englewood Cliffs, N.J., 1976, Chapters 4 and 6.

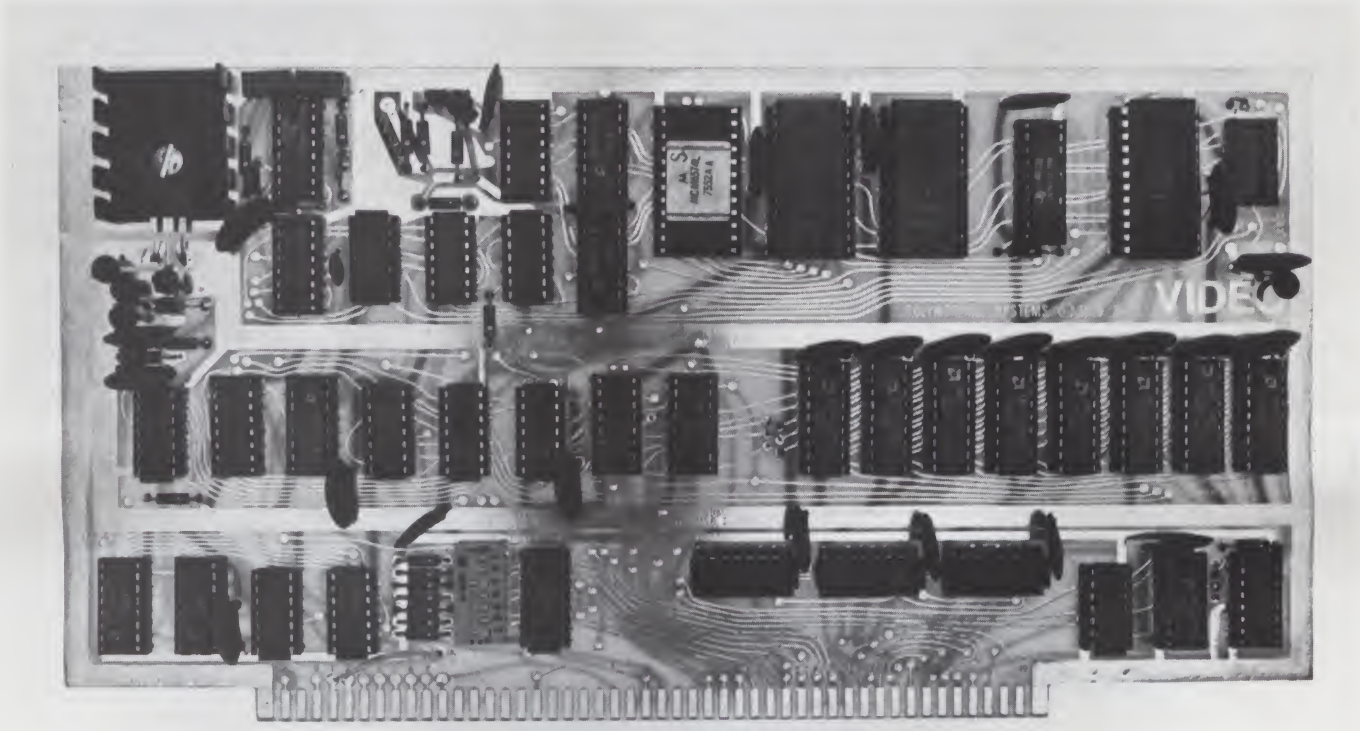
Rony, P., Larsen, D. and Titus, J., *The Bugbook III*, E and L Instruments Inc., Derby, Conn., 1976.

Mitchell, E. H., "In's and Out's of Computers for Beginners", *Popular Electronics, Vol. 9 No. 6* (June, 1976), pp. 47-54.

Rony, P., Titus, J., and Larsen, D., "Komputer Korner", *Radio Electronics*, Vol. 47, No. 4 (April, 1976), pp. 18-30.

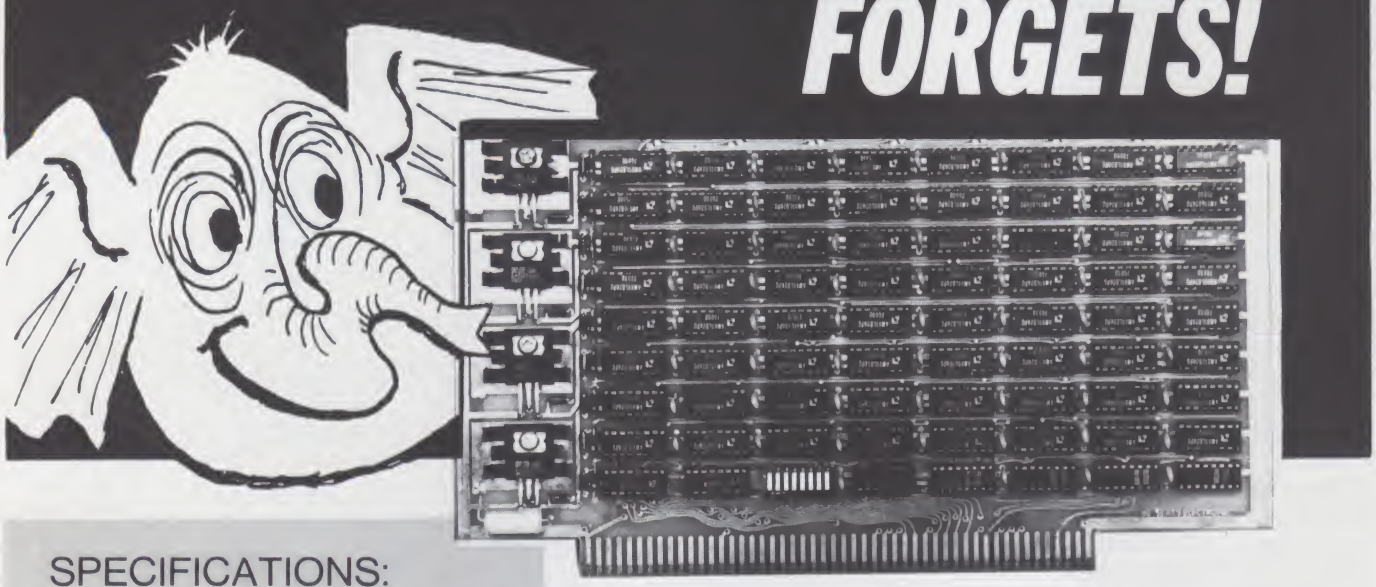
"The Smart Machine Revolution", *Business Week*, July 5, 1976, pp. 38-44.

Mallon, M., "Computer Construction Hints", *Interface*, August, 1976, pp. 52-53.



A Video Interface Board (Courtesy of Polymorphic Systems).

The first 8-K that NEVER* FORGETS!



SPECIFICATIONS:

8K SC - 8 Specifications:

Access Time:	500 ns Max. (225 max on request)
Current Req:	Less than 200 ma per 1024 words maximum
Memory Chip:	AMD 91L02 APC **(low** power 1K x 1)
Voltage Supply:	+5 to +10 volts
Battery Standby:	1.5 to 2 Volt, Automatic power loss sensing circuit. Elliminates need for switches.
Address Select:	8 ea. Spst. switches in a Dip IC package (No longer any need for a soldering iron to change address.)
+5 Volt regulated:	4 ea. 7805 regulators with individual heat sinks to run cooler.
Wait States:	NONE! Your wait light will not burn because of a memory wait state.

ALL ADDRESS, CONTROL, AND DATA OUT LINES FULLY BUFFERED

Circuit Board

Double sided, G10 glass epoxy board
Plated through holes. 5 mil. tin minimum
Solder reflow processed
Solder mask on both sides of PC board
Component lay out silk screened on component side of PC board
Gold plated edge contacts
No jumper wires used
Professional layout techniques used

ALL ADDRESS, CONTROL, AND DATA OUT LINES FULLY BUFFERED

•

QUANTITY, DEALER, AND CLUB INQUIRIES INVITED

•

PLUG IN-COMPATIBLE WITH ALTAIR" AND IMSAI®
IC SOCKETS INCLUDED

$295.00 Kit — $394.00 Assembled — $2.00 Shipping and handling

LIMITED TO CAPACITY OF STANDBY BATTERY

SEALS ELECTRONICS®

BOX 11651, KNOXVILLE, TN. 37919
(615) 693-8655/TWX 810 583 0075

DEALERS & DISTRIBUTORS*

Mr. Peter Bickerdike
CHANNEL RADIO & ELECTRONICS
18 East Ortega Street
Santa Barbara CA 93101
Phone: 805-965-8551

THE COMPUTER MART
314 5th Avenue
New York NY 10001
Phone: 212-279-7757

THE DATA DOMAIN
111 South College
Bloomington IN 47401
Phone: 812-334-3607

THE COMPUTER MART
625 W. Katella Avenue, #10
Orange CA 92667
Phone: 714-633-1222

* COMPUTER MART
DISTRIBUTING COMPANY
Orange CA
714-633-4634

THE COMPUTER MART
1097 Lexington
Waltham MA 02154
Phone: 617-890-0677

THE COMPUTER MART
151 Kline Blvd.
Colonia NJ 07067
Phone: 201-574-2173

* HOBBYTRONIC DISTRIBUTORS
1218 Prairie Drive
Bloomington IN 47401
Phone: 812-336-6380

* MJB RESEARCH & DEVELOPMENT
36 W. 62nd Street
New York NY 10023
Phone: 212-245-8530

* MCED COMPANY
Suite 101, 1600 Hayes Street
Nashville TN 37203
Phone: 615-329-1979

Computer Control of the World!

... turning ac powered devices on and off with your computer

Hey, I think this is one a lot of us have been waiting for! Just the thing for those of us who would like to be able to turn the world on and off with our home sytems. Since the world seems to be running on 120 V ac, Chris has come up with an interesting article telling us how to turn it on and off . . . using an optoisolater. – John.

Chris Bowick
Box 35482
Georgia Tech
Atlanta GA 30332

Optoelectronics is fast becoming an important area in the design of all types of computer control circuitry. There are many different types of optoelectronic devices and even more possible uses for them. One of the more interesting devices in this class of electronics, which is becoming popular in computer interface applications, is the Photon-Coupled Isolator, or Optoisolator. With this amazing device it is now possible to control high voltage high power circuitry with TTL voltage levels from your personal computer system. This is done without actual electrical connection between the two devices. Therefore the line voltages, and their associated noise components, are not allowed back into the logic.

This article will be a short look at only one of the many General Electric optoisolators available today, the H74C1. I will first try to cover some very basic theory on the device and then throw a couple of experiments at you for some "hands on" experience.

Theory

The H74C1 contains a gallium arsenide infrared emitting diode (IRED) and a light-activated silicon controlled rectifier (LASCR) in a 16-pin dual in-line package. The IRED is nothing more than a light emitting diode (LED) which emits light in the infrared region. The optoisolator is specifically designed to work with the 7400, 74H00, and 74S00 series of logic gates to control 120 or 240 V ac power devices.

The IRED is a very interesting device in that its electrical characteristics are very similar to any run-of-the-mill silicon diode. One unique difference with the IRED, however, is its ability to emit infrared light when it is forward biased. This characteristic is utilized in the H74C1 by coupling the light through some transparent dielectric material to a LASCR (or photo SCR, as it is sometimes called).

Since the photo SCR and almost all other silicon light detectors are nothing more than a photodiode junction (not to be confused with the LED) and an amplifier, our best bet for understanding the photo SCR could quite possibly be found in understanding the photodiode. Does that seem logical?

The photodiode is a *reverse biased* P-N junction device which allows current to flow when light of the proper wavelength and intensity is directed toward the P-N junction (see Fig. 1).[1] Here's how. In any good reverse biased semiconductor diode, a depletion region will be formed around the two P-N type materials. This depletion region is, of course, nothing more than an electric field inside the diode. If light is directed toward the P-N junction, the energy from the light causes hole-electron pairs to be created, which are swept across the junction by the electric field that the depletion region had set up in the first place. This causes current to flow in the external circuitry of the diode.

The photo SCR works in a similar manner. It does, however, have one more terminal than a regular photodiode. This terminal is called the gate. For simplicity's sake let's just say that the gate allows the circuit designer to vary the sensitivity of the SCR for each different application. The gate is normally connected to the cathode of the photo SCR through some value of resistance. The particular value depends on the given design specifications.

Now that we understand the basic operation of the IRED and the LASCR as separate entities, let's see what happens as the two are merged into one discrete unit. When this is done (see Fig. 2), it becomes obvious even to the most casual observer that a truly amazing device has been generated! When the IRED is forward biased it

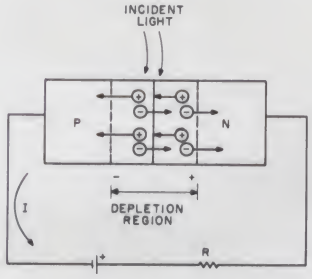


Fig. 1. Incident light on a reverse biased photodiode causes current flow.

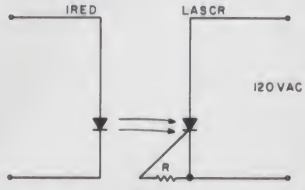


Fig. 2. The IRED and LASCR integrated into one discrete unit. R is an externally connected resistor.

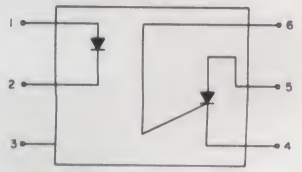


Fig. 3. Pin diagram for the H74C1. Pin 3 has no connection.

Infrared Emitting Diode	
Power Dissipation	100 milliwatts
Forward Current (Continuous)	60 milliamps
Forward Current (Peak, 100 micro sec. 1% duty cycle)	1 ampere
Reverse Voltage	6 volts
Photo SCR	
Peak Forward Voltage	200 volts
RMS Forward Current	300 milliamps
Forward Current (Peak, 100 micro sec. 1% duty cycle)	
Surge Current	5 amperes

Table 1. H74C1 specifications.

emits light. This light is coupled through the transparent dielectric material to the P-N junction of the LASCR, causing it to conduct during each half cycle that it is reverse biased. This produces a current in the external circuitry of the photo SCR giving us complete optical isolation between 120 V ac and the TTL voltages which turned the IRED on in the first place.

"So how can I use it?" you say. Well, why don't we take a look!

Technical Information and Experiments

All technical specifications for the H74C1 optoisolator are available from the General Electric Semiconductor Products Department in Syracuse, New York. Some of the more important specifications have been listed for your convenience in Table 1, and the pin diagram for the six pin DIP is shown in Fig. 3.

As stated previously, the optoisolator can be used as a TTL interface to line voltage operated devices. In Fig. 4, which is a very simple example of this type of operation, it is apparent that a high logic input (+5) would not allow the IRED to conduct. Since the LASCR would not "see" any light, it would remain off and the 15 Watt bulb would therefore not be lit. If, however, a low logic level were applied to the input, the IRED would conduct (because it is now forward biased), emitting infrared light and gating the LASCR on during its reverse biased cycles. The bulb would then light! Well what have we here? Could it be a 15 Watt logic indicator? Of course, the bulb will glow with only half the normal brightness due to the LASCR conducting during only half of each line voltage cycle. As you can see, the concept is really very simple.

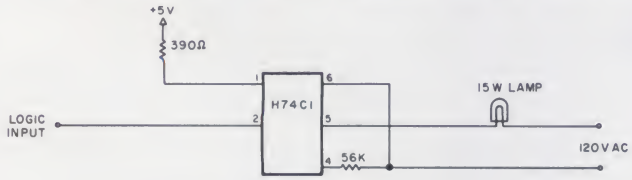


Fig. 4. A 15 Watt logic probe?

Another very simple example you might like to try is seen in Fig. 5. This circuit will accept BCD data and then select any one of ten different loads to be switched on (be sure that the load you use doesn't cause the current and power ratings of the SCR to be exceeded). The LED display is there merely to indicate which load has been selected. The function table and pin diagram for the 74145 are seen in Fig. 6. It is obvious from the function table that for any given valid input, all of the 74145's outputs are high except one. That one output corresponds to the BCD data you place at the inputs to the chip. For example, if the D, C, B and A inputs were 0, +5, 0 and 0 volts respectively (corresponding to the number 4 in binary notation), then the outputs of the chip would have 0 volts at output number 4 and +5 volts at all other outputs. Therefore, with the H74C1 operating as described before, load number 4 would be "on" while all others would be "off." The LED display would, of course, show the number 4 in this case, indicating which load had been turned on. By changing the BCD data at the inputs to the 74145, we can therefore select any one of the ten loads we wish to be turned on.

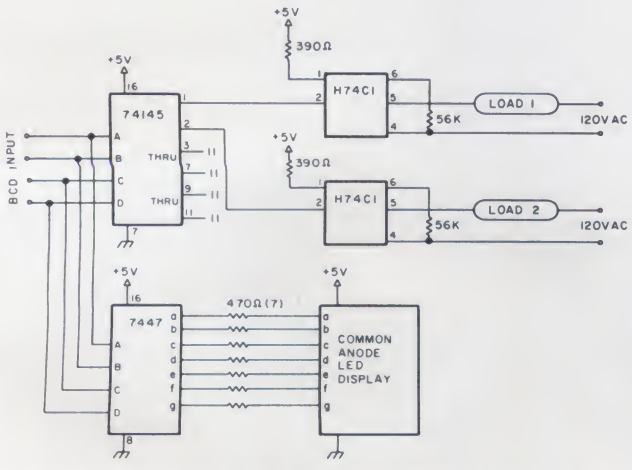
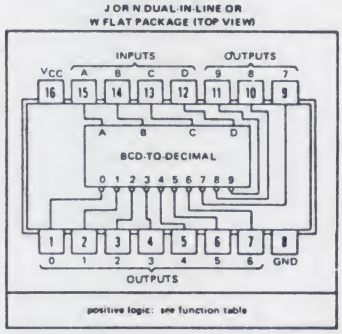


Fig. 5. Choose any one of up to ten different loads using this setup. The LED display will indicate which load is on. Note that pins 3 through 7 and 9 through 11 of the 74145 are terminated in the same fashion as pins 1 and 2.



FUNCTION TABLE

NO	INPUTS D	C	B	A	OUTPUTS 0	1	2	3	4	5	6	7	8	9
0	L	L	L	L	L	H	H	H	H	H	H	H	H	H
1	L	L	L	H	H	L	H	H	H	H	H	H	H	H
2	L	L	H	L	H	H	L	H	H	H	H	H	H	H
3	L	L	H	H	H	H	H	L	H	H	H	H	H	H
4	L	H	L	L	H	H	H	H	L	H	H	H	H	H
5	L	H	L	H	H	H	H	H	H	L	H	H	H	H
6	L	H	H	L	H	H	H	H	H	H	L	H	H	H
7	L	H	H	H	H	H	H	H	H	H	H	L	H	H
8	H	L	L	L	H	H	H	H	H	H	H	H	L	H
9	H	L	L	H	H	H	H	H	H	H	H	H	H	L
INVALID	H	L	H	L	H	H	H	H	H	H	H	H	H	H
	H	L	H	H	H	H	H	H	H	H	H	H	H	H
	H	H	L	L	H	H	H	H	H	H	H	H	H	H
	H	H	L	H	H	H	H	H	H	H	H	H	H	H
	H	H	H	L	H	H	H	H	H	H	H	H	H	H
	H	H	H	H	H	H	H	H	H	H	H	H	H	H

H = high level (off), L = low level (on)

Fig. 6. Pin assignments and function table for the 74145.

Conclusion

The optoisolator is a truly fascinating device; it's versatile, fun to work with, and, most important, inexpensive. The cost usually runs much less than $2.00 from most General Electric distributors, and this is a lot less than its interface *counterpart (the* solid state relay) which usually sells at several times this price.

Hopefully, through reading this article, you have been introduced to something both interesting and useful to you and your home system. I have only scratched the surface concerning the usefulness of this device, but who knows, maybe I have opened a few doors.

I would like to thank Ms. Maureen Brown and Mr. David Eversman for their help, suggestions, and criticisms in preparing this article. ■

References

1. *General Electric Optoelectronics Manual*. W. H. Sahm, Semiconductor Products Dept., Syracuse NY 13201.
2. *The TTL Data Book for Design Engineers*, First Edition. Texas Instruments Incorporated.

Dennis Brown
President, Wave Mate
1015 W. 190th St.
Gardena CA 90248

Wire Wrapping

... Try it!

Dennis Brown is the man responsible for the Jupiter II computer system by Wave Mate, and, since the system modules are built using wire-wrap construction, it would be safe to assume he is a strong advocate of wire-wrap. And, so he is. His article discusses the advantages and disadvantages of going the wire-wrap route and also introduces us to a universal wire-wrap board offered by Wave Mate. A layout diagram for the board is included for those who would prefer to make their own. — John.

*Wire-wrap is a registered trademark of Gardner-Denver.

Photos by Roger Young.

This is the first in a series of articles on the Wave Mate Jupiter II computer system. The purpose of this article is to lead you slowly through the wire-wrap* maze and give you a better understanding of wire-wrapping. More than this, I hope to calm your fears about wire-wrapping and show you the advantages it offers. Perhaps you may even see the beauty in this jungle of wires. I'm also going to introduce you to, and point out the benefits of, a truly beautiful and universal wire-wrap card.

What is Wire-Wrapping?

Wire-wrapping is a solderless technique, for assembling electronic circuits, in which a small wire is wrapped tightly around a post, usually square in shape. Bending the wire around the sharp corner of the post crushes the oxide layer on both the wire and the post together, creating an oxide free, gas tight, metal-to-metal contact. The wire also stretches as it is wrapped over the post, causing the post to be slightly twisted and under constant pressure at the points of contact. The posts are normally gold plated to prevent oxidation, and the wire is silver plated copper for better conductivity.

As the contact ages, the silver atoms of the wire and the gold atoms of the post fuse, creating an extremely reliable contact. This connection is immune to shock, vibration, time, humidity and

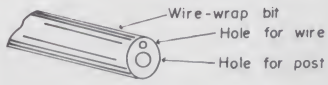


Fig. 1. Detail of wire-wrap bit.

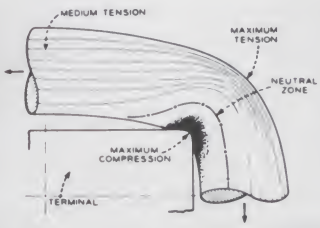
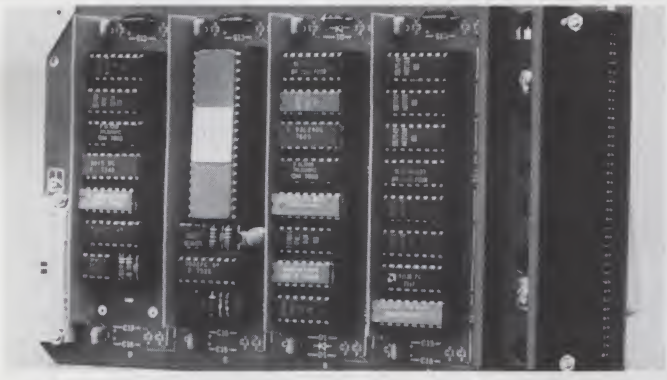


Fig. 2. Detail of wire biting into wire-wrap post.



CPU module for the Jupiter II computer system built on Wave Mate's universal wire-wrap cards.

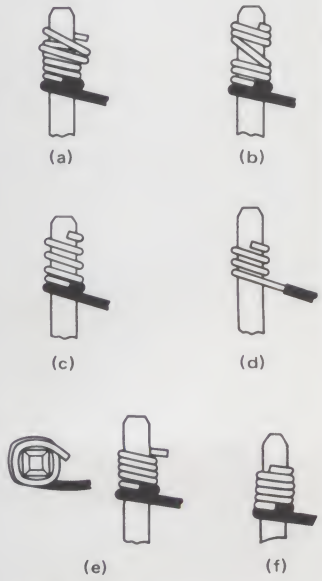


Fig. 3. Troubleshooting bad wraps. (a) Overwrap caused by pressing too hard on the tool. This condition is impossible if a power tool with a backforce spring is used. (b) Spiral wrap caused by uneven pressure on the tool. (c) Open wrap caused by no pressure on the tool. (d) Insufficient turns caused by not pushing the stripped end of the wire far enough into the tool. (e) Pigtail caused by using the incorrect tool size. For 30 gauge wire use a tool specified for 30-32 gauge. (f) Correct wrap. The post should have at least one turn of insulation.

temperature changes. A wire may be removed and another wrapped on the same post with the same high quality connection as the first.

How is Wire-Wrapping Done?

There are many tools available to make wire-wrapping easy, including hand and electric tools as well as air powered models. However, all are basically the same, consisting of a hard steel shaft (called a bit) with one large hole in the center of the end for the post and a smaller hole near the edge for the wire (Fig. 1).

A wire with one inch of insulation stripped from each end is pushed into the small hole so that the insulation is slightly inside the tool. The tool containing the wire is placed over the wire-wrap post and turned by hand or power, with a slight downward pressure. The loose end of the wire should be held stationary, forcing the wire to come out of the tool and wrap around the post so that the corners of the post bite into the wire forming a gas tight connection (Fig. 2.) With a powered tool, the bit turns at high speed and is surrounded by a stationary sleeve to avoid damage to nearby wires.

For most electronic projects, the wire-wrapping posts should be 0.025 square and 0.625 inch long. The wire should be 30 gauge Kynar insulated solid copper wire, silver plated, and stripped one inch at each end. The wire-wrapped bit should be rated for 30-32 gauge wire. If these specifications are used, the bit will wrap about one turn of insulation and seven turns of wire on the post.

Wire-wrapping is easy, even if you have never done it before. After a few minutes of practice, you should get a good consistent wrap every time. When wiring on the second level of a post, be sure to leave space between the top of the first wrap and the start of the second wrap. If you are having trouble getting consistent wraps, refer to Fig. 3 for help.

In contrast to other types of construction, wire-wrap offers many advantages and few disadvantages. Compared to soldered point-to-point wiring, wire-wrap wins hands down. Wire-wrap connections are much easier to remove and many times more reliable.

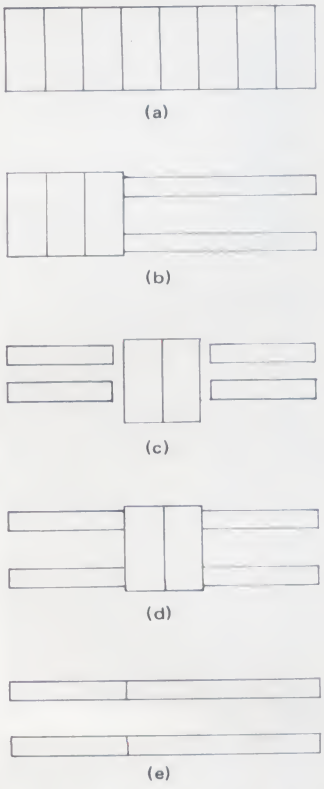
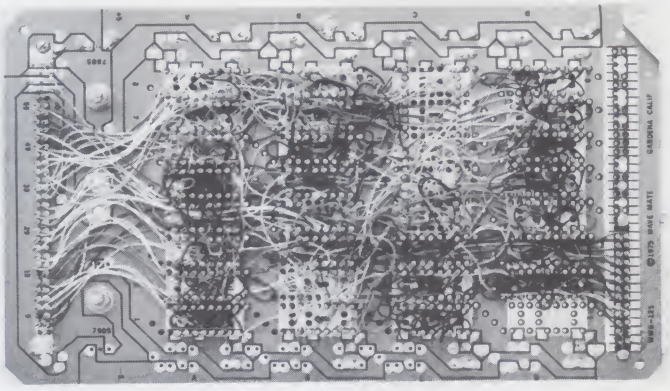


Fig. 4. Detail showing possible socket configurations of any one of four rows. (a) Eight 18-pin sockets. The sockets are the tabloc variety which needs no soldering or adhesives to be retained in the card. Other screw down or adhesive-mount types can be used. (b) Three 18-pin sockets and one 40-pin socket. The 40-pin socket is made from two 20-pin strips which are held to the board by soldering to two isolated lands. (c) Two 18-pin sockets and two 22-pin sockets. (d) Two 18-pin sockets and two 24-pin sockets. The 24-pin socket is made from two 12-pin strips which are held to the board by soldering to two isolated lands. (e) One 24-pin socket and one 40-pin socket, or one 64-pin socket.



Wire-wrap side of CPU module.

Printed circuit cards are the closest competitor to wire-wrap. For high volume applications, PC cards are more economical to produce. A single sided PC card has the most rugged construction, but few electronic projects are simple enough for one and most usually require the use of two or more layers of printed circuitry. If a multilayered PC card is accidentally flexed the connections can be damaged. The etched layers will be pulled away from the plated-through-holes that are required to interconnect them. A wire-wrap card, made with a single sided PC card, may be flexed vigorously without damage.

Wire-wrapping forces you to use IC sockets, which make diagnosing and repairing your project a snap. You can damage integrated circuits by soldering them directly to a PC card; diagnosis and repairs then become extremely difficult and time consuming.

Wire-wrapping allows integrated circuit packages to be spaced much closer together than PC cards. Therefore, interconnections are much shorter, resulting in cleaner wave forms and reducing noise in the system. Since all the interconnections are wired, a ground plane can be etched on the card to eliminate noise and crosstalk, further increasing system reliability.

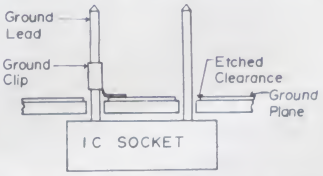


Fig. 5. The ground clip slides over the wire-wrap post. The leg is then soldered directly to the ground plane. The PC card has an etched clearance around each wire-wrap post to prevent accidental shorting of signal leads.

The most important advantage of using wire-wrap is the ease and speed with which you can go from idea to finished project. Wire-wrap also enables you to change and upgrade your circuits with ease. *Often you can go* from start to finish of a project in a single day.

All of these sound like great advantages that everyone can benefit from, so why isn't everyone selling wire-wrapped computers? The answer is twofold. First, the cost of a wire-wrap card is higher than an etched PC card because each IC must be socketed and wiring a board takes longer than wave soldering. Second, all of these advantages are theoretical in nature. Most designs using wire-wrap cards fall short of the ideal for one or more reasons. Therefore, it is



Fig. 6. Detail of bus bar used for power distribution. The bar is made from 0.025 thick half hard brass, with gold or tin plated leads.

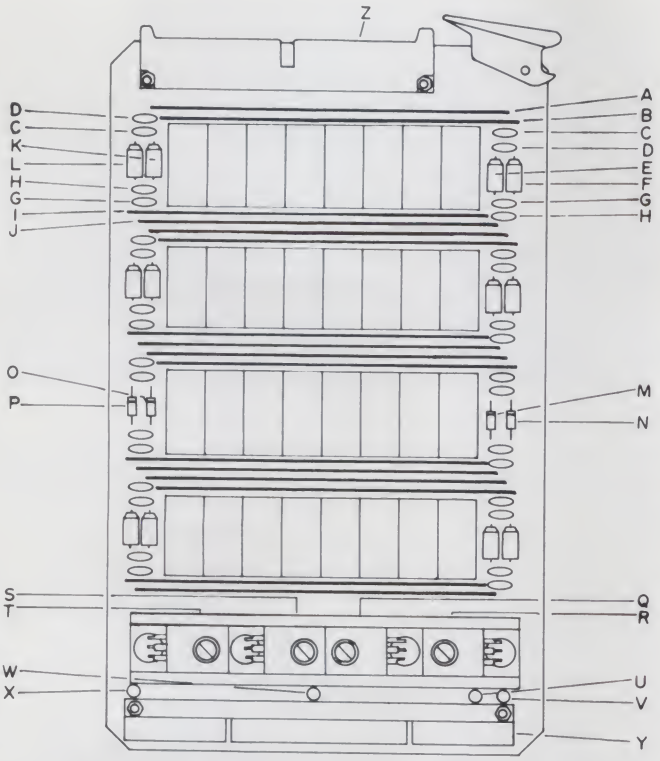


Fig. 7. Maximum possible component layout for universal wire-wrap module: (a) Row D +12 V bus bar. (b) Row D +5 V bus bar. (c) Row D +5 V high frequency bypass capacitor (0.1 uF). (d) Row D +12 V high frequency bypass capacitor (0.1 uF). (e) Row D +12 V low frequency filter capacitor (15 uF). (f) Row D +5 V low frequency filter capacitor (15 uF). (g) Row D -5 V high frequency bypass capacitor (0.1 uF). (h) Row D -12 V high frequency bypass capacitor (0.1 uF). (i) Row D -5 V bus bar. (j) Row D -12 V bus bar. (k) Row D -12 V low frequency filter capacitor (15 uF). (l) Row D -5 V low frequency filter capacitor (15 uF). (m) +12 V reverse voltage prevention diode (1N4001). (n) +5 V reverse voltage prevention diode (1N4001). (o) -12 V reverse voltage prevention diode (1N4001). (p) -5 V reverse voltage prevention diode (1N4001). (q) +12 V power regulator (7812). (r) +5 V power regulator (7805). (s) -12 V power regulator (7912). (t) -5 V power regulator (7905). (u) +12 V input filter capacitor (1 uF, 35 V tantalum). (v) +5 V input filter capacitor (1 uF, 35 V tantalum). (w) -12 V input filter capacitor (1 uF, 35 V tantalum). (x) -5 V input filter capacitor (1 uF, 35 V tantalum). (y) 72 pin bus connector (Berg R/A hdr shld 2x36 30AU). (z) 50 pin I/O header (3M #3496-3005).

important, if you are seriously considering building or buying a wire-wrap project, to examine closely some of the features of the wire-wrap card that can make it a dream machine or a nightmare.

What Features Should a Wire-Wrap Card Have?

To take advantage of all the new large scale integration (LSI) chips in your computer system you will need cards that can hold a wide variety of socket sizes: 14 to 18 pins on 0.3 inch centers, 22 pins on 0.4 inch centers and 24 to 64 pins on 0.6 inch centers. To power these new chips, the cards must have provisions for distributing up to four voltages, usually ±5 and ±12 V. Sometimes other voltages are necessary, including -9, -3 and +26 V.

The cards should have a ground plane; every IC socket pin should be capable of connecting directly to the ground plane without using a wire-wrap wire. All voltages should have provisions for high frequency bypass capacitors every few packages. If on-card voltage regulation is used, a large heat sink should be provided.

The cards should be large enough to hold a CPU, a memory or an I/O controller but small enough to ensure the modularity of the system. There should be room for many I/O connection pins and bus connections.

The Wave Mate Universal Wire-Wrap Card

Over the past few years Wave Mate has succeeded in making a universal wire-wrap card which meets all of the above specifications. We now manufacture all of our plug-in modules using these cards.

Every Wave Mate plug-in module uses the same printed circuit card. However, each module contains a unique combination of integrated circuits and other components. The standard card is designed to accommodate the maximum number of components required to implement any function. Thus, it is unlikely that any module would require the board to be completely filled with components.

The printed circuit board is designed to accommodate four rows of sockets. If the sockets contain up to 18 pins, they are mounted vertically, providing capacity for 8 sockets per row for a maximum of thirty-two 18-pin sockets per board. See Fig. 4(a). (22-, 24- or 40-pin sockets are mounted horizontally.) Each 24-pin socket displaces three 18-pin sockets in a row; a row may contain two 24-pin and two 18-pin sockets or one 24-pin and five 18-pin sockets. See Fig. 4(d). A 40-pin socket displaces five 18-pin sockets. See Fig. 4(b).

Each socket location is designated by a row letter (A-D) and a column number (1-8).

All integrated circuits used on the card are plugged into sockets. The sockets perform the functions of both holding the ICs and providing a wire-wrap post for each pin of an IC. Discrete components are also plugged directly into wire-wrap sockets, making layout extremely versatile.

Except for the edges, the entire etched surface of the card is a ground plane with etched clearance around each post (Fig. 5). All voltages are bussed along the edges of the card. The printed circuit card is capable of providing four different voltages to each socket mounted on the card. Each voltage is transmitted to the sockets by a power bus that spans each row. See Fig. 6. The power bus for a voltage is only installed on a row if that voltage is required by an IC on that row.

A set of 1 uF bypass capacitors filter the input to each of the regulators. Up to three 15 uF tantalum low frequency bypass capacitors are provided for each voltage used on the card. Two 0.1 uF high frequency bypass capacitors are used at the end of each bus bar. A reverse-polarity-prevention diode is added for each voltage on the card.

The card may connect to the system bus through a 72-pin connector. Its gold pin type contacts provide a higher reliability connection than a PC edge connector. Up to 60 pins of flat cable I/O connectors can be placed at the top of the card. A card ejector can be installed at the top of the card for easy removal from a card cage. Fig. 7 is an illustration of the universal wire-wrap board.

The Wrap-up

The advantages and disadvantages of both wire-wrapping and printed circuits have been shown along with an explanation of the wire-wrapping process. From this you can see that wire-wrapping, as a viable method of producing high reliability equipment, offers flexibility and ease of construction.

The universal wire-wrap card presented here overcomes the problems associated with most other wire-wrap cards. It enables you to realize the full range of advantages available with wire-wrap.

Future issues of *Kilobaud* and other magazines will be offering many construction articles, some for PC cards and others for breadboard-type construction. The universal card described here can be used conveniently with many such projects. The universal wire-wrap module is available (either in kit form or assembled), along with a complete line of wire-wrap tools and accessories, from Wave Mate, 1015 W. 190th St., Gardena CA 90248.

Next month we'll begin a discussion of the Wave Mate Jupiter II computer system, which is built using the wire-wrap cards presented here. ■

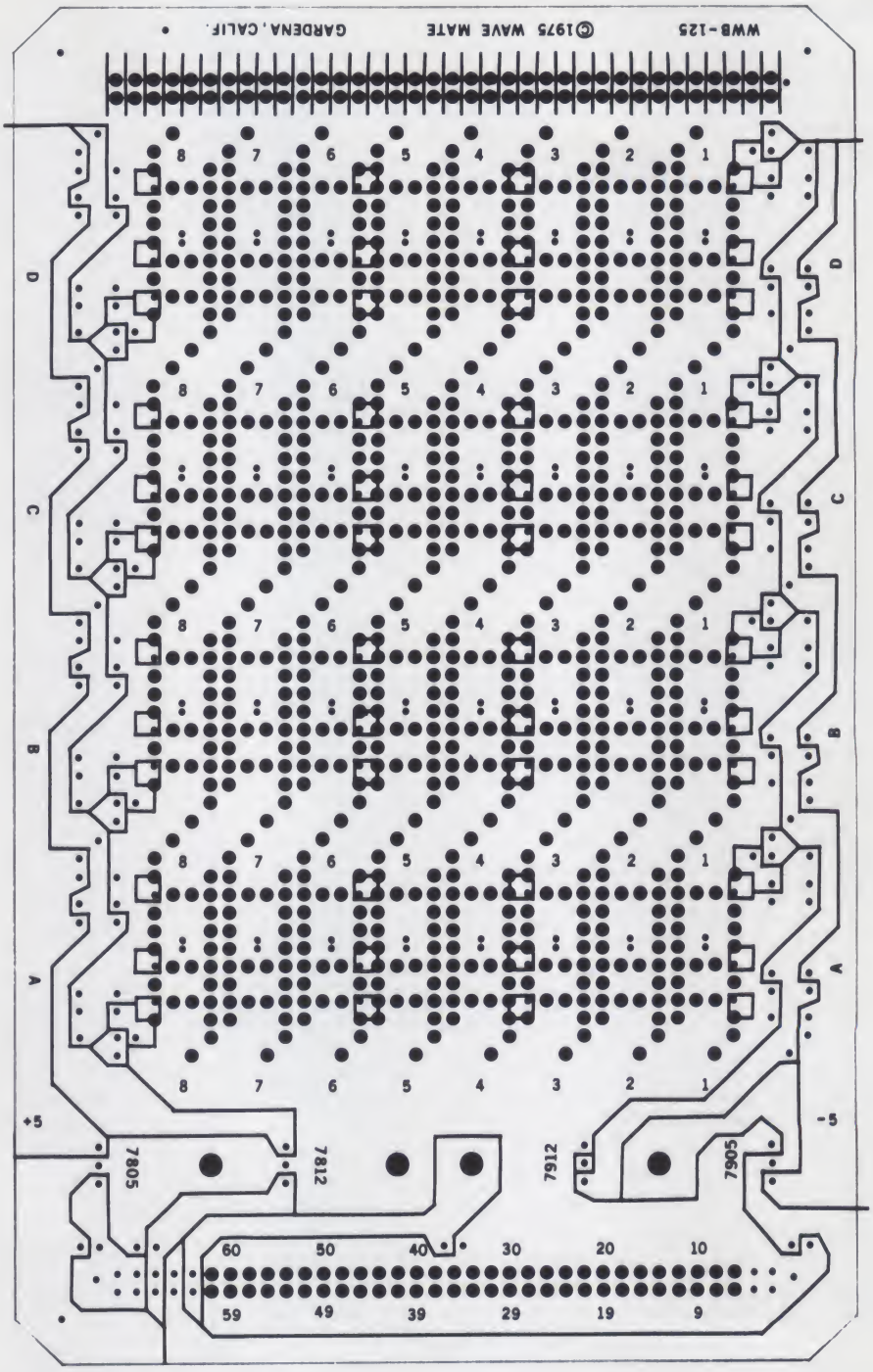


Fig. 8. Full size layout for the universal wire-wrap PC card. The black lines are etched away, the white areas are copper. Two oz. copper plated glass perforated board may be used by the more adventurous hobbyist who wishes to make his own. All the parts and designs shown are copyrighted and may not be manufactured for sale or distribution.

Letters to the Editor

from page 10

thing. I know nothing about hardware!

I read with interest Wayne Green's editorial reprinted in *PCC Newspaper*. As a systems analyst in the U.S. Air Force and with a software background, including a degree in computer science, I know the value of having ready-to-use software. *I hate to reinvent the wheel!* everytime I want to do something new in the way of software programs. Hope your suggestions works out.

Ronald G. Martin
Omaha NE 68123

Books on Basics

Certainly you should start your new mag for the beginner, but you must prepare for the time when a great number of the readers are more advanced. All articles of a fundamental nature should be slated for eventual inclusion in books. Also, I really do hope we can get a little math into *Kilobaud*. Computer approximations are very important but cannot be understood *at all* without a little math.

Webb Simmons
San Diego CA 92111

You've brought up some good points, Webb. There are going to be newcomers getting into personal computers all the time, either as users or hobbyists, hopefully, in large numbers. Kilobaud *will have a wide variety of fundamental material (so we won't be repeating ourselves) and, at the same time, we'll provide interesting and* provocative *material for the non-beginner! With regard to books . . . you hit it right on the head. We're going to be putting out some dandies. And as far as the math is concerned . . . again, you're right in there. A "little" math in* Kilobaud *should be just fine. While most of* Kilobaud's *readers can use simple high school algebra and trig, most of them probably don't want to. They're going to prefer practical circuits or practical approaches to a subject. – John.*

Readers' Service

Glad to see that you will be going through with another computer-related publication. As the title shows,

continued on page 90

Dick Wilcox has come up with a machine-independent discussion on the design of a "home brew" OPERATING SYSTEM. Dick recently completed the hardware and software development of a hobbyist 16-bit microprocessor board, which plugs into the Altair bus (would you believe?) and is complete with a sophisticated BASIC and Operating System. He has some very good ideas to share with those of us interested in developing our own operating systems. We'll be hearing a lot from Dick. After getting the "big picture" in this first article, we'll find the details on developing the individual modules in future articles. (And, I would suspect we'll also be seeing a future article on that 16-bit machine.) — John.

Dick Wilcox
1342 Mauna Loa Rd
Tustin CA 92680

The Hobbyist's Operating System

... Part 1: Introduction and Master Plan

OPERATING SYSTEM: A program, or collection of routines, whose primary purpose is to assist in the running and monitoring of other programs. An operating system usually doesn't perform the desired specific task (which will be accomplished by the program being executed), but rather performs sub-tasks which enable the program to operate more efficiently. The main program in an operating system (the one that's running the show) has been referred to by a variety of names, such as *monitor, executive* and *supervisor*. In general, all terms mean the same thing and are sometimes used interchangably, although there are the purists who differentiate between them by the magnitude of functions which they perform. In this instance the term "operating system" usually refers to the larger and more flexible program which truly performs the task of *monitoring* the actual running of the user program, as well as the loading of it. MONITOR would therefore seem to be a good choice when referring to the operating systems main, or control program.

There are two main thoughts that I try to keep in mind whenever I design a new function into a system, or especially when I sit down to initially rough out a new system approach from scratch. First, the main task that is given to an operating system of any worth is to give a good, but easily understandable, command language to the person using the computer.

Consider, for instance, the following two commands:

```
R DO,CMPLR,1530
*DSK:PROG.LST/L,PROG.BIN=DSK:PROG.SRC

COMPILE PROG WITH LISTING
```

Both command sequences will get the job of compiling the program and creating a list file but it is quite obvious which one will cause fewer premature gray hairs. Understandable commands to your computer certainly may not be the highest priority on your list if you are a high school student trying to impress the girl next door, but for the hobbyist who is designing a system for his wife to use, simplified language can be a real boon. It just might even help justify an additional 16K of memory instead of that new set of cookware which, as any devoted computer hobbyist will attest, has very little socially redeeming significance. The second command sequence shown is an ideal that probably will not be obtainable, for now, in the average microprocessor system due to the large amount of code required to process this "free-form" style. In actual practice, we shall probably end up with a command language which is somewhere between the above.

The second point I consider is the assistance that the operating system gives the programmer in the development and debugging of new programs. The ease of editing and saving source files, assembly or compilation of these files, and the eventual debugging of the object program can sometimes save several times over the effort of doing the same tasks on a limited system. The amount of effort expended in making the operating system a powerful tool should be proportional to the expected use of the system for new program creation and test. A system which is eventually slated to be used mostly for production runs will not justify using resources for program development. On the other hand, a system which will spend most of its time in the hospital recovering from programming blunders should have as many gimmicks as possible to assist the doctor (you). A speedy recovery makes for a ready patient upon which you can test your next diabolical new coding. The system which I am proposing here can eventually help you go from editing to assembly to test and results in a couple of minutes.

The operating system presented here, and the basic ideas behind the functions it will perform, is of a magnitude somewhere between the small monitors presently available on the hobbyist market and the large systems that run on the godlike computers in commercial operations. In its fully expanded form it will provide you with an extremely flexible command language including the ability to create your own commands without regenerating the system itself. It will be floppy-disk based although it will run without a disk in a limited form. It will contain generalized I/O facilities which means you will not have to rewrite your programs each time you add a new piece of gear to your computer. All data will be handled in the form of named files on disk or other storage media and will be processed through the operating system routines. Multi-user time

sharing in a limited environment will be discussed for those who wish to delve into this frustrating field. These are the main functions of any good system but by no means should be considered either necessities or restrictions.

One point I would like to make at this time is that although the general ideas presented here are used in many large-scale systems, they do lend themselves readily to the microcomputer field within the constraints of limited resources. The fully expanded operating system which I am describing has been implemented on at least three different makes of computers in some form or another. I am currently implementing it on a microcomputer system, and I project that the totally resident portion of the monitor will take up about 12K bytes of memory. This article is not intended to be a theoretical dissertation on large system practices. It is to be treated as an introduction to the techniques that make flexibility in operating system design for microcomputers feasible — well within the grasp of the hobbyist's abilities. In writing about something as technical as software design, it is almost impossible to remain generalized enough at a low-key level and still convey sufficient information to be useful for design. Thus, I will attempt to first give a broad overview of each new concept introduced and then expand upon that concept at a technical level in enough detail to give the interested reader a solid basis for design.

Obviously there are tradeoffs to be considered when designing your own system. Defining what your system is to do, and to what depth, is the first step in the process. Do not overlook the fact that an operating system is probably the most flexible and dynamic piece of software you may ever write, so be sure to keep in mind the areas that you may want to add to later on as your system progresses. A little bit of planning at this stage will result in many saved hours of rewrite later on. Start off small, but keep big ideas (and ideals) tucked away for ready reference in the coding of your system.

Shared Routines

Any operating system worth its weight in salt must perform certain functions during the process of loading and running the problem programs. These functions will include, at minimum, interaction with the operator's terminal, I/O processes to read and write the program storage device, and some minimum conversion routines for decimal- or hexadecimal-to-binary formats. Since these routines are necessary for proper operation of the operating system, why not make provisions for them to also be accessible to the program that is to be run? You don't make your friends bring along their own dishes when they come for a visit, do you? It makes as little sense to force each program to bring along its own I/O and conversion routines when these same routines are sitting idle in the resident monitor.

Now we are faced with the problem of how to efficiently utilize these routines. Two main factors must be considered here. The use (access) of these routines should be kept simple in order to discourage the programmer from generating them in his own program. An I/O call which takes a lot of thinking just to get a character printed on a terminal is defeating its own main purpose: to make the programmers job an absolute joy (well . . . at least maybe not so formidable). The use of these routines must also be implemented in such a way as not to consume a lot of memory space or execution time — an overhead that cannot be tolerated in any system.

It should be obvious that in order to use these monitor routines the program must either know exactly where they are or else be able to reach them through a general monitor call by passing along a code to indicate the specific routine that is desired. The general pros and cons to both methods will be discussed briefly. The first method, that of knowing the exact location of each routine, is the fastest in terms of execution speed since there is no wasted effort by the monitor in decoding a specific code and then executing the routine. The major drawback here is that, each time the monitor is altered and the routines change position in memory, all the calling programs must be reassembled to reflect the new routine addresses. This can be circumvented by using a table of calling sequences to the routines, where the table never changes position in memory. This method places constraints on the monitor design and also is not easily implemented in some microprocessors. The second method has the distinct advantage that once the routines are assigned specific codes or numbers, they can then be called by these numbers which will never change, even though the monitor may go through several gyrations during its lifespan. The main disadvantage with this scheme lies in the fact that each time a routine is called there is an extra penalty which must be paid in execution speed because the monitor itself must decode the assigned routine number and then transfer control to that routine.

Another advantage to this second method is sometimes overlooked because of its subtle implications. Most of the routines which will be implemented as shareable units will require one or more arguments to be passed to them for use in the specific execution of the routine itself. These arguments are variables such as device names for I/O functions, data to be read or written, numbers to be converted, or addresses of data in memory to be acted upon. As these routines in the monitor become more sophisticated, the need to evaluate and preprocess these arguments into a suitable form rapidly increases. If the arguments follow a predefined format and the preprocess function is the same for all the routines to be called, this action can be accomplished by the general monitor call just prior to transferring control to the desired routine. In the operating system which I have developed, I have chosen to implement the second method. Future articles will expand upon the various methods which may be employed to implement this coding scheme in the more common microprocessor systems of today.

Major System Units

In the above section I presented the idea of incorporating the ability for user programs to call some of the functions which were inherent in the operating system monitor because the monitor needed these routines in order to perform its own chores of loading and running programs. In a more flexible system, we can expand on this idea and include routines in the monitor which, though the monitor doesn't use them, are commonly used by user programs. If the calling structure is defined properly at the onset, all that is required to add a new routine to the system is to assign it the next available code number and then add it to the monitor program. Of course, there are tradeoffs to be considered here which exist in any generalized application. The more nifty little routines that we include in our monitor, the easier it is to sit down and write a new program to run under that system because much of the

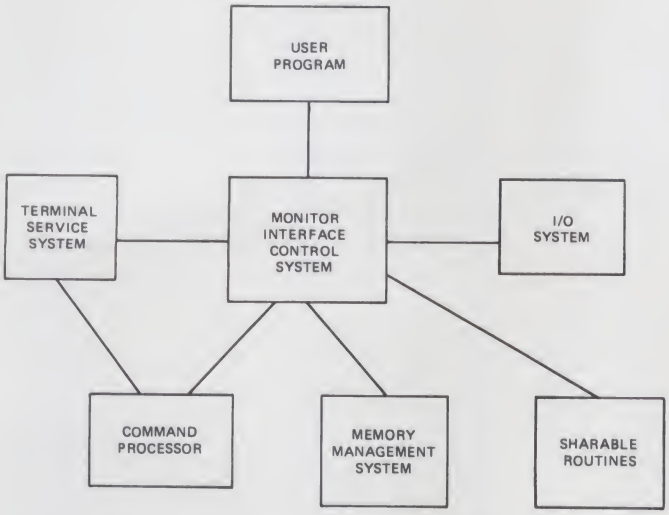


Fig. 1. Expanded operating system block diagram.

work is already done for us. However, keep in mind that these little devils subtly gobble up valuable memory resources whether they are actually put to use by each program or not. An operating system that does all kinds of great things but takes up 31K bytes out of a 32K byte computer might be as useful as a gold-plated garbage can.

Following is a list of the major system components which I have found to be of sufficient value to justify their existence in an operating system such as we would like to develop. (Refer also to Fig. 1.) This list is neither complete or necessary but merely is a guideline which you can build upon or set an eventual goal for your system. Each item in the list should be reviewed with care and evaluated as to its particular worth for you. The goal in this evaluation need not be whether to include the major unit, but rather the degree to which you will implement it and the proposed subunits or options it will support. Remember that vanilla and chocolate ice cream both sell equally well because no two people are alike.

1. Terminal service routines for communicating with the variety of available video displays and hard-copy devices

2. Logical I/O routines to relieve the programmer of communicating with peripheral devices on a physical transfer level

3. Command level processor to evaluate operator input commands and perform specific system functions or execute a user program

4. File-structured disk management system for storage of programs, source files, and data files on floppy or hard-surfaced disks

5. Numeric conversion routines to convert ASCII numeric input to packed binary format and vice versa (decimal, octal, or hexadecimal)

6. Memory management system for allocating and keeping track of user requests for main memory areas and program usage

7. Monitor subroutine dispatching system for linking monitor routines to the running program for execution (discussed previously)

We will briefly describe here each major unit as to its general function and the tradeoffs involved in implementing it in various stages during operating system development. Future articles will treat each item in greater detail giving the reader the necessary information to evaluate its implementation into his or her system.

Terminal Service Routines

With the possible exception of a few fanatical diehards, nobody likes to control his system by flipping a few toggle switches up and down in some ancient ritualistic rhythm. The big boys use a terminal to control all functions performed by the computer and if we intend to play the big boys' games we should strive to play in full dress uniform. The operating system should accept commands from the operator via one or more terminal devices. Processing input from any terminal requires some special considerations when compared to input from other peripheral devices such as paper tape or cassette readers. Special editing features, such as character rubout and full line erase, must be incorporated to make the operators life more bearable. Most terminals require that the input character be transmitted to the corresponding output port (*echoing*). This allows you to view what you have typed in and, if you type like I do, this is an absolute must! The variety of terminals available to the hobbyist also poses some other interesting problems.

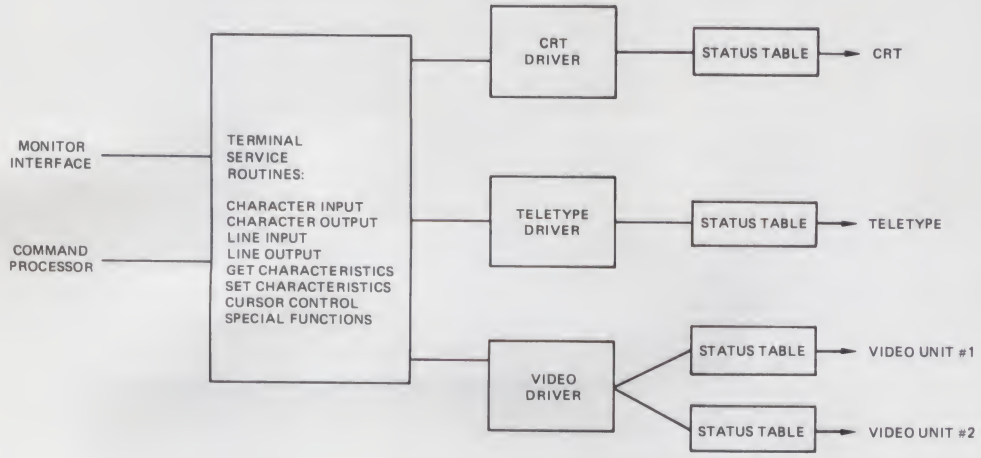


Fig. 2. Typical terminal service system.

GODBOUT

BILL GODBOUT ELECTRONICS
BOX 2355, OAKLAND AIRPORT, CA 94614

TERMS: Add 50¢ handling to orders under $10. Cal res add tax. No COD; to place BankAmericard® or Mastercharge® orders, call (415) 562-0636, 24 hours.

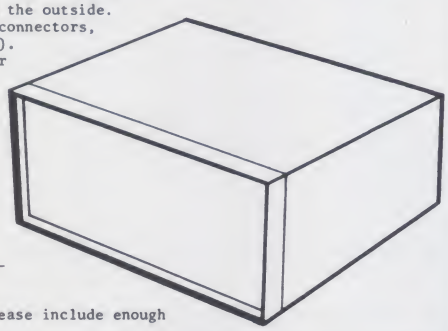
OUR FLYER HAS IT!

If you want to see the latest in interesting goodies for computer hobbyists, ask for our current flyer. It has Altair/IMSAI etc compatible peripherals; some of the newest octal parts, a complete line of TTL, CMOS, and Low Power Schottky ICs; power supplies; connectors; enclosures; and all the capacitors & resistors you could ever want, as well as a full line of Vector products---SEND FOR OUR FLYER!!

MENTION THE NAME OF THIS MAGAZINE ON YOUR ORDER AND YOU CAN DEDUCT 5% OF THE TOTAL!

NEW! Enclosures from VECTOR

Deluxe cases that look beautiful---no screws or fasteners mar the good looks from the outside. Interior slots hold card guides, connectors, etc. (not included with enclosure). All enclosures available in either dark blue or black with white front panel. Order from the following---



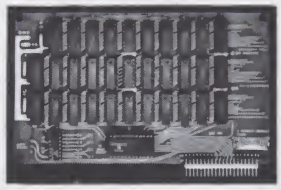
#VP5-17-17U: 5.51" H, 17.58" W, 17.1" D. **$79.25**

#VP7-17-17U: 7.26" H, 17.58" W, 17.1" D. **$84.00**

#VP9-17-21U: 9.01" H, 17.58" W, 21.6" D. This is exactly the same size as the IMSAI 8080 microcomputer. **$96.50**

All units shipped unassembled; please include enough for postage (excess refunded).

BASIC RAM $88



This 4K by 8 memory board has no frills, just storage. Designed for compatibility with JOLT systems, this board is also ideal for other microcomputers using bi-directional buss systems. Same size as JOLT memory card, plus lo-power operation (1.3Amax) to keep you on good terms with your power supply.

If you don't need the onboard regulation or address / data buffers of our "Bigger Brother" 4K board, then this is the way to go.

Sockets included for all ICs.

9601 FACTORY FRESH PRECISION ONE-SHOT 4/$1

FACTORY FRESH PRECISION ONE-SHOT 4/$1
FACTORY FRESH PRECISION ONE-SHOT 4/$1

ECONORAM™ in kit form $99.95

We took everything we learned from selling 4K x 8 RAM boards for the past year, added some of this year's circuit tricks, and came up with ECONORAM --- a memory board that is even more remarkable due to its low price. We've engineered this with the user in mind, giving you several benefits:

* 3 regulators to share power load, plus optimized thermal design, means a cooler running microcomputer
* Typical current consumption of under 750 ma gives your power supply a break
* Fast --- Zero wait states
* All TTL support ICs are latest Low Power Schottky types
* For reliable and unambiguous data transfer, all addresses, data lines, and outputs are buffered for minimum loading and maximum output capability
* Power-on clear included

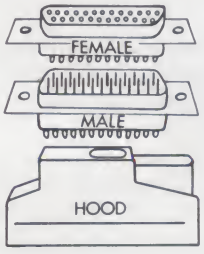
All these features are packed on to an Altair-sized, industrial quality double-sided PC board, with sockets for all ICs, 7 tantalum capacitors for power supply decoupling, and plenty of bypass capacitors---39 of them, in fact, as well as a logic print and instructions.

also available assembled $129.95

Our popular **ECONORAM 4K x 8 RAM** board is now available **assembled, tested, and warranteed for one year.** Plug it in to your Altair or IMSAI and enjoy the same performance that has made the kit such a success---guaranteed zero wait states and current drain of 750 mA or less; on board regulation, easy address selection, and lots more.

Submini "D" Connectors!

MALE WITH HOOD $3.95; FEMALE CONNECTOR $3.95.



lo profile sockets

HERE'S YOUR CHANCE TO BUY SOME TOP QUALITY SOLDERTAIL SOCKETS AT THE RIGHT PRICE.

14 PIN...10/$1.95	16 PIN...10/$2.15
22 PIN...10/$3.50	20 PIN...10/$3.10
36 PIN...10/$5.50	24 PIN...10/$3.60
	40 PIN...10/$6.15

LTD QTY SPECIAL: 18 Pin 8/$1.50

COOL MEMORY

DISCOUNT: Buy 10, take 10%
Buy 100, take 20%.

We just got a batch of **low power** and **fast** (450 ns maximum--full temp range) 2102L1 static RAMs. Perfect for memory boards--and at our usual reasonable price: **$1.95!**

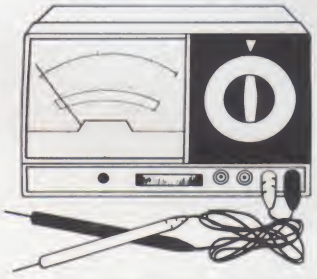
CHIP SET $36.95

1-8080A, 1-8224, 1-8212, AND AN 18 MHZ CRYSTAL AT A SAVINGS OVER INDIVIDUAL PRICES: FORMS THE CORE OF A QUALITY 8080A CPU BOARD.

Multi Meter $14.99

An outstanding buy...6 DC Volt ranges, 3 DC current, 5 AC Volts, 2 ohms, and dB scale too. 50 uA movement; plus "off" position damps meter for safe transportation. Includes 34 inch test leads but less 1 AA battery. Small--a compact 5¼" x 3¼" x 1 5/8".

Please add $1 postage and handling.



.01% ACCURACY IN A TO-5 CAN...not just useful for fixed frequency oscillators, since most numbers can be used with simple binary dividers to generate other frequencies.

For example, 15.360 can divide down to 60 Hz for timing pulse purposes;

16.384 divided by 2^{14} gives 1 Hz timing pulses; and so on. All frequencies $4.95 --- apply IC discount schedule for quantities. ALL FREQUENCIES IN KHz.

STATEK CRYSTALS

Choose from:

10.000	12.800	15.360
16.000	16.384	
18.641	19.200	20.480
		30.720
31.500	32.768	
38.400	40.960	60.000
76.800	100.00	153.60
	240.00	

SENTRY CRYSTALS--$4.95 PER CRYSTAL **CRYSTALS**

Series mode, fundamental, wire leads, for hams and computer bugs. Choose from 4 MHz (PACE clock crystal), 5 MHz 8 MHz, 10 MHz, 12 MHz, 15 MHz, 18 MHz (8080 clock crystal), and 20 MHz.

"EconoRom" $179.95

ALTAIR 8800/IMSAI PLUG-IN COMPATIBLE. This is a 4K by 8 EROM board...the ideal place for putting software, be it assembler, editor, or any custom set of routines. Additionally, this board may expand to 8K x 8 by simply adding more sockets and EROMs; also available is a 2K x 8 version if you don't need a full 4K. LOW POWER: 8K board requires ½A @ 5V, & 150 ma @ -12V. Buffered addresses for lightest loading, buffered outputs for maximum drive. Kit includes sockets, double-sided quality PC board, on board regulators, logic print, and instructions. Program it yourself, or have us do the programing.

8K X 8 BOARD $269.95 **2K X 8 BOARD $135.00**

8080 Software Board $189.95

We took our ECONOROM board kit, but instead of including blank EROMs, these are programmed with assembler, editor, & monitor routines for the 8080. This is a valuable first step if you're trying to get away from machine language programing. There's not really enough room here to fully describe all the functions of the software...but if you send us $2.95 (refundable with order), we'll send you our software packet that includes instructions listing, schematic, and assembly data.

Many of the terminals require special functions to effect proper line-oriented output displays.

All these variables increase the desirability of a single terminal input service routine built into the operating system. If all user programs request terminal input via this monitor routine then, if the user ever adds a new terminal device to the system (see Fig. 2), he merely has to change the one handler. As the operating system gets more sophisticated, the terminal service routine can be expanded to include multiple-device handlers, interrupt driven input systems, and control of multiple jobs in a time sharing environment. The limit of expansion is up to you. In any case, even the most basic operating system requires the handling of input from your terminal and since this is a reasonably complex procedure, it should be made available for other programs to share. Also, if you get involved in the automatic command processing system described in a later section, you will need to funnel all terminal input through a common routine for proper operation.

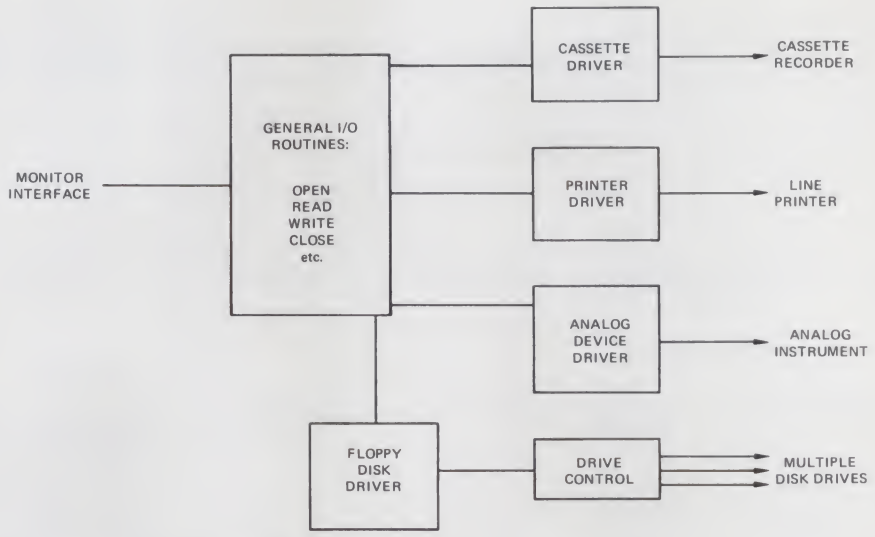


Fig. 3. Typical I/O System block diagram.

Logical I/O Routines

Most programs written for data handling require the data to be read and/or written from one or more peripheral devices. On your system you may start out with naught but a lonely terminal but eventually you will move up to the big time with cassettes, floppy disks, and all sorts of other diabolical devices *designed* to shuffle bits around in and out of your central processor. The variety of schemes which must be used to untangle and make sense out of these bits and pieces is enough to put all but the most stalwart of enthusiasts away where white coats are the fashion of the day. If you go through these gyrations each time you write another program, you either are trying out for the "Harrassed Man of the Year" award or you have not yet heard of the concept of centralized logical I/O processing. This is merely a fancy buzzword for a very simple and practical approach to handling these various peripheral devices. You design a single set of monitor calls such as OPEN, READ, WRITE, and CLOSE and use these calls in your programs each time you wish to read or write data from somer peripheral device (see Fig. 3). You inform the monitor which device you wish to use (there are various ways to do this) and the monitor I/O routines do all the dirty work.

Sound simple? Well, almost but not quite. Many of the more sophisticated devices have certain funny little quirks built into them, such as floppy disks which require more detailed information to be given before the specific data can be located and processed. Disks in particular must be structured in some logical manner to allow the data to be stored in little chunks commonly referred to as *files*. These files should be referenced by some meaningful name instead of an absolute location on the disk (and so the plot thickens as the file structure develops). The program should be written to process data from the most complex device that is expected to be used; and then, the monitor should be designed to recognize the fact that the simpler devices (such as a paper tape reader) do not need these additional arguments and to discard such devices. Since this is a complete subject in itself it will be dealt with in more detail in a future article. The main point to be remembered here is that by starting out with some predefined I/O structure, no matter how limited it may be initially, additional devices may be added to your system by merely writing a new monitor driver (routine) to interface to that device. All programs which have been written using your I/O calls can then access the new device without alteration.

Command Level Processor

An operating system is only as good as the assistance it can provide to the person using it. A way must exist for the system to fully understand what is expected of it if it is to perform properly. The orders that are given to a computer are called *commands* and can range from a meaningless string of special characters designed to appease the machine's thirst for complexity to a set of straightforward English words which get the job done. Again, as always, there are tradeoffs to be considered. The evaluation of single characters is a lot simpler than the evaluation of complete words which may vary in length and position. The obvious tradeoff in the other direction is that words are a lot easier to understand than special characters, especially if you are not used to talking in gibberish. The functions to be performed by the operating system will also contribute greatly to the design of the command language. If the command words are to be stored in some sort of table within the monitor, they should be as concise as possible (which heads us right back in the direction we are trying to avoid). Also, if we use specific commands to perform internal system functions, the monitor itself must be regenerated each time a new command is implemented.

The method which I am about to propose is not totally unique in itself but does have some interesting points which make it worthwhile. Instead of having a set of specific commands which perform internal system functions, why not just treat each command as the name of a program and have the monitor load and then execute that program? The the commands themselves need not be stored within the

monitor, and the implementation of new commands merely involves the creation of the program to perform the command function. This method is most advantageous when used in a disk-based operation since the programs can be stored on the disk and loaded at high speed by the monitor routines. I have also used the same idea with systems that did not have a disk but, instead, treated main memory as the program storage area. It was then up to me to ensure that the program being executed was loaded (from cassette, paper tape, etc.) by the monitor before executing the command which invoked it. Such a system must allow multiple programs to be held in memory at one time in order to be effective.

Disk Management

Keeping track of programs and data which are stored on floppy disks is an entire science in itself. Data stored on a cassette is written as a single stream of sequential characters from start to finish either in one chunk or a series of chunks referred to as *records*. Reading this data back into the computer merely involves backing up the tape to the beginning and then reading the records in the same order in which they were written. Data stored on a disk, however, can take on several different formats and be accessed in a number of ways, depending on the application in use. Reading data from disk does not set the limitation of retrieval in the same order as it was written. You also have the ability to select only portions of data and write or rewrite without changing all the rest of the records associated with it. A disk can be likened to a slower extension of main memory in some of its capabilities and therefore will require some special routines to efficiently store and retrieve data from it.

For the sake of clarity, I will direct my next statements toward the most popular devices currently available to the hobbyist — the floppy disks. Data is read and written on these disks in fixed-length records, each record containing 128 bytes, this collection of records (called a *file*) can consist of an entire program to be executed or of a series of ASCII characters which comprise the source program before assembling or compiling. The file might also be just a random ordering of data used by some of the programs in any format the programmer desires. Since each disk can store around 2000 records (each 128 bytes long), it would be a waste of space if a system only allowed one file to be on each disk at any one time. There are many popular schemes available for the ordering and accessing of these files on the disk and each has its merits for particular applications. The tradeoffs which the hobbyist must make in order to implement an efficient, yet inexpensive, system (memory-usage wise) will be discussed in a future article dedicated to file organization techniques for microprocessor systems. Since this series of articles is aimed to an eventual disk-based system, most of the ideas presented will be centered around the use of the disk for program and data file storage. The reader who is not yet into disk systems can still use many of the ideas presented here for manipulating files in main memory; he will then have a better understanding of large system techniques when he eventually moves up to a disk-based operating system.

Numeric Conversion Routines

One of the areas that the novice who starts to write his first monitor may overlook is the incorporation of numeric conversion routines into his monitor which can be used by all running programs. Here cleverness and forethought can really save both development and execution time. Almost all numeric data is stored and manipulated in binary within the computer yet must be accepted and displayed to the outside world in a familiar format. This format may be decimal, for the most common applications, or octal or hexadecimal, for the exotic system debugging that invariably entices everyone into its trap. After all, what fun is a computer if you can't jump into its weird world of funny numbers once in a while?

The process of converting from one numeric format to another can take on many coding forms and is an area where the programmer can express his innovative abilities. The duplication of this effort into each and every program soon becomes boring, however, and you will soon find yourself looking for a way around the problem of inputting and outputting numeric data. If these routines can be put into the monitor and then called by the programs, the time saved in the development of new programs will be significant. The question is, how many different routines should we incorporate, and how fancy do we make each one? The decision is yours, and it will depend on the frequency of usage and the importance you place on the formatting capabilities of the data routines. Such frills as leading and trailing zero suppression, decimal point handling, and space positioning are neat, but they do require memory, of course. Normal usage will probably dictate that the fancy frills, if any, be incorporated into the decimal data routines and the straightforward "no frills" method be used for octal or hexadecimal conversions. If you insist on working with funny numbers, you probably won't mind roughing it for awhile.

Memory Management

Any task that is to be performed by the computer will require the use of main memory for program execution and immediate data manipulation. The easier the operating system can make the task of managing the available memory resources, the easier it will be for you to develop new programs. The operating system must have a few basic routines for its own use so that programs can be loaded into their proper places and work buffers can be allocated for system functions (see Fig. 4). Routines which inform the user program of where the available memory is located and how much memory is free for its use are helpful, especially in systems where the amount of available memory changes on a regular basis. More sophisticated techniques can be devised that allow the program to request a specific amount of memory for work space and allow the monitor to allocate that memory wherever it happens to be free at the time.

There are many techniques that can be employed for the efficient management of main memory resources and most of the decisions in this area will be based upon the type of hardware in use. Certain processors inherently have more flexible instruction sets which lend themselves to fancy memory allocation schemes. Probably the most easily adaptable systems are those processors that allow various degrees of instruction relocatability. This is another fancy term which merely means that the processor has been designed so that either some selected portion or all of the instructions may be freely moved about in memory without the need to reassemble the program to execute it. Indexing and PC-relative techniques are the tools which many manufacturers use to implement some level of instruction relocatability and the method that you employ in your operating system and to what level you employ it will depend greatly on this. This is another arena

in which the creative gladiator will eventually come out on top.

Keep It Simple

There are no secret weapons to use in the design of a monitor nor are there any giants to be killed in one fell swoop. A good system will normally start out as a few simple routines that are necessary to get the basic job done and then expand gradually as the wizardry and courage of the programmer grows. To start out with a design that incorporates all the frills that could possibly be desired in an operating system is inviting frustration and possible failure. An operating system should grow in small steps like a stamp collection. In turn, this will give a continuing flow of creative feedback which never hurt anyone. I hope to be able to give sufficient inspiration in the areas of operating system design. The objective is to encourage experimentation in this fascinating field rather than set down a few hardfast techniques which are to be considered the best for everyone. And by all means, let's not forget the cardinal rule of any hobby: Share your knowledge with others. There are very few ideas which I would call entirely my own while keeping a straight face. I continually give open thanks to all those who have gone before me and contributed to the background that allows me to twist, bend, fold, staple (and sometimes mutilate) the techniques I now use freely. ■

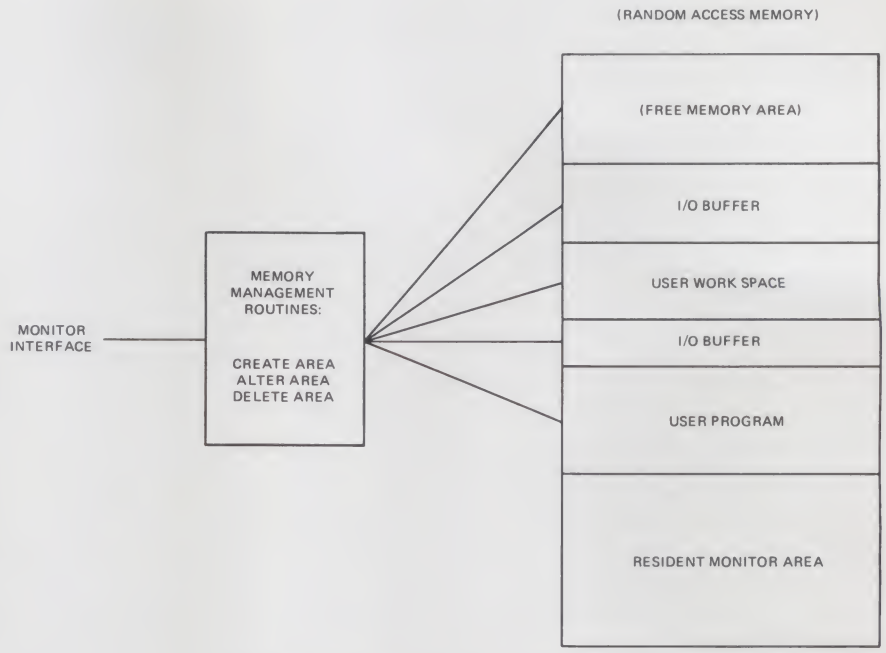


Fig. 4. Memory Management block diagram.

a full range, 5-function 3½ digit multimeter



Data Precision Model 134

SALE PRICE

$169.00

RANGES
DC VOLTS: 0 to ±1.999/19.99/199.9/1500 Volts
AC VOLTS: 0 to 1.999/19.99/199.9/1000 Volts
DC CURRENT: 0 to ±1.999/19.99/199.9/1999 mA
AC CURRENT: 0 to 1.999/19.99/199.9/1999 mA
RESISTANCE: 0 to 199.9 Ω through 19.99 Megohms (6 ranges!)
ACCURACIES
(See Specifications, back page, for detailed accuracy statements)
DC VOLTS: ±0.2% F.S. ±0.2% of reading
(1000V range: ±0.5% F.S. ±0.5% of reading).
AC VOLTS: ±0.7% F.S. ±1% of reading
(1000V range: ±0.7% F.S. ±2% of reading).
DC CURRENT: ±0.5% F.S. ±0.8% of reading
(1000 mA range: ±0.5% F.S. ±1.8% of reading).
AC CURRENT: ±0.5% F.S. ±1.3% of reading
(1000 mA range: ±0.5% F.S. ±2.3% of reading).
RESISTANCE: ±0.5% F.S. ±0.8% of reading
(10 megohm range: ±0.5% F.S. ±1.8% of reading).

AST/SERVO SYSTEMS, INC.
20 REPUBLIC ROAD NORTH BILLERICA, MASS. 01862
617-667-8541



Cabinets clockwise from top: CPU, Dual-cassette drive, Keyboard, 9" Monitor.

Meet The Digital Group

If you are seriously considering the purchase of a microcomputer system for personal or business use... or just beginning to feel the first twinges of interest in a fascinating hobby... the Digital Group is a company you should get acquainted with.

For many months now, we've been feverishly (and rather quietly) at work on our unique, high-quality product—a microcomputer system designed from the inside out to be the most comprehensive, easy-to-use and adaptable system you'll find anywhere. And our reputation has been getting around *fast*. In fact, you may have already heard a little something about us from a friend. We've found our own best salesmen are our many satisfied customers.

There's a good reason. Simply, the Digital Group has a lot to offer: state-of-the-art designs, a totally complete systems philosophy, unexcelled quality, reasonable software, affordable prices and the promise that our products will not become rapidly obsolete, even in this fast-moving, high-technology field.

The Advantages

Here are a few specific advantages of our product line:

- We offer interchangeable CPUs from different manufacturers (including the new "super chip"—the Z-80 from Zilog) which are interchangeable at the CPU card level. That way, your system won't become instantly obsolete with each new design breakthrough. The major portion of your investment in memory and I/O is protected.
- Digital Group systems are complete and fully featured, so there's no need to purchase bits and pieces from different manufacturers. We have everything you need, but almost any other equipment can be easily supported, too, thanks to the universal nature of our systems.
- Our systems are specifically designed to be easy to use. With our combination of TV, keyboard, and cassette recorder, you have a system that is quick, quiet, and inexpensive. To get going merely power on, load cassette and go!
- Design shortcuts have been avoided—all CPUs run at full maximum rated speed.
- All system components are available with our beautiful new custom cabinets. And every new product will maintain the same unmistakable Digital Group image.

The Features

Digital Group Systems—CPUs currently being delivered: Z-80 by Zilog 8080A/9080A 6800 6500 by MOS Technology

All are completely interchangeable at the CPU card level. Standard features with all systems:

- Video-based operating system
- Video/Cassette Interface Card
 512 character upper & lower case video interface
 100 character/second audio cassette interface
- CPU Card
 2K RAM, Direct Memory Access (DMA)
 Vectored Interrupts (up to 128)
 256 byte 1702A bootstrap loader
 All buffering, CPU dependencies, and housekeeping circuitry
- Input/Output Card
 Four 8-bit parallel input ports
 Four 8-bit parallel output ports
- Motherboard

Prices for standard systems including the above features start at $475 for Z-80, $425 for 8080 or 6800, $375 for 6500.

More

Many options, peripherals, expansion capabilities and accessories are already available. They include rapid computer-controlled cassette drives for mass storage, memory, I/O, monitors, prom boards, multiple power supplies, prototyping cards and others. Software packages include BASICs, Assemblers, games, ham radio applications, software training cassettes, system packages and more (even biorhythm).

Sounds neat—now what?

Now that you know a little about who we are and what we're doing, we need to know more about you. In order for us to get more information to you, please take a few seconds and fill in our mailing list coupon. We think you'll be pleased with what you get back.

the digital group

P.O. Box 6528
Denver, Colorado 80206
(303) 777-7133

OK, I'd like to get to know you guys better.
Send me the whole package!

Name _____

Address _____

City/State/Zip _____

Solving Some of the Software Interchange Problems

We're going to make the Kilobaud *Software Library work. In the short time it will take to get the thing going into full swing, we're going to be publishing a series of articles dealing with the software interchange standards this effort will require. Naturally, we'd be very interested in reader responses to these articles and the ideas they put forth. Jef Raskin emphasizes the interchange of BASIC programs in the following article and touches upon assembly language programs also. He has some points which make such good sense your first reaction is, "Well of course that's the way it should be." I'm sure he's developed these ideas from his many years of playing with these monsters. Aside from being an ex-Computer Center Director at UC San Diego, he has taught courses in programming (of course), computer animation, and computer music. And, would you believe that his basement contains an Altair 8800, an IMSAI, an Apple, and a Poly 88? I have this strong feeling we're going to be hearing more from him. — John.*

Jef Raskin
Box 511
Brisbane CA 94005

When you receive a program from another personal computer user or from a manufacturer, more often than not it will require modification before it will run on your system. The wide variety of configurations and choice of input and output devices make it impractical to write a program that will run on any system without modification. That goal is seldom obtainable because, if it had to have routines to drive every kind of device known, even the simplest program would become too large to fit into most computers. Furthermore, any enterprising company could obsolete the software by merely inventing a new I/O device. And, it seems this happens nearly every day. The large computer manufacturers try to solve the problem with "do everything" operating systems. But they don't mind selling megabyte memories to their customers, and the customers (usually not knowing how much of their memory is being wasted) think that the large system software's size is giving them much more power. This is a trap *we* can't afford.

Fortunately, it is possible to set up standards, all of which are generally considered good programming practice, that make it easy to interchange any program from one machine to another. There are some limitations: If the program is in assembly language then, of course, the two machines must have the same instruction set. (Although, as we will see, programs can be written in such a way as to facilitate interchange between different instructions sets.) If the program is in a higher level language, then machine dependent tricks (such as PEEK, POKE, INP and OUT in MITS BASIC) cannot be used. (Later on we will detail ways in which they can be used and still maintain ease of transfer.)

I/O Configuring

If your program is designed to work *only* with a *particular* I/O device then say so in your program writeup. But most programs will work equally well with a Teletype, Flexowriter, CRT terminal or Video Display device. The fundamental rule to ensure this capability is that *the main program must do no I/O whatsoever.*

All input and output should be done from subroutines. For any device at most four subroutines are required. A set is written for *each* device.

1. **START.** This subroutine does any initialization required by the device. Many serial ports with a UART require initialization. Some cassette interfaces require a start byte and/or a sync byte. You often have to throw away a random character sitting in an input or output buffer and, in some cases, a delay may be needed for a device to come up to speed. Anything that must be *done once* before a device can be used for input or output is done in this routine.

2. **GET.** For devices that send bytes to the CPU only. This routine, when called, returns one byte of data from the input device. Any handshaking, waiting, or control codes that must be *set up for each byte* are done in this routine. As far as the main program is concerned this routine is a magic box that delivers one byte when called. A protocol for determining

when a device is done must be established. Usually a final delimiter in the input stream or an initial byte or two giving the length of the information to follow will do the trick.

3. **PUT.** For devices that receive bytes from the CPU. This routine, when called, sends one byte to the output device. Any handshaking or timing or control codes that must be *sent for each byte* is done in this routine.

4. **FINISH.** Occasionally, a device must be turned off after it is used. This routine will perform that function when necessary.

An analogous set of routines is required for some mass storage systems, such as disks, except that in this case a block of memory rather than a single byte is the burden of the routine. None of these I/O conventions will be of any use whatever, though, unless the programs are well documented. Documentation is another distinct area which needs minimum standards badly.

When a program is obtained and, if the above standards are adhered to, the user will merely (ha) have to find (from the accompanying documentation) where the I/O subroutines are in the program. If the program was written correctly *there will be no other I/O routines in the program* and there will be *only one* START, GET, PUT, and FINISH *for each device.* These small subroutines will then have to be modified or replaced with routines designed for the user's I/O gear.. This will be especially easy if no "hard" device codes or control or status bytes are used. That is, if instead of a constant being used in an I/O instruction, a variable is used whose value is set in the START program. This is good programming practice in any case. The same applies to addresses. "Soft" or relocatable code (unfortunately not available on some of the rather primitive assemblers) makes customizing easier.

Avoiding Tricks

We all love to be clever in writing programs. And anybody who has programmed for a while can come up with some neat "hacks" that do an operation in an unexpected way. Of course, being a good programmer you realize that someone reading your program won't take the time to figure out what you have really done. They will simply shake their heads in wonderment and proceed to modify the program in such a way that it will never work again! Since the aim of programming is not to impress your peers but to create something that works and is reliable (yes, software *can* be made to be reliable or unreliable just like lawnmowers or refrigerators), the programmer should avoid tricks. If a "trick" seems to give your program a decided advantage, then be sure to *fully* explain what you have done. One byte saved by a clever trick will usually require three or four *lines* of explanation. Don't assume the readers of your program will know what you were thinking. They won't. This is true in higher level languages, such as BASIC, but is especially true in assembly language. Remember that it is easier, given good documentation, to recreate a program than it is to create the documentation given the program!

Interchangeable BASIC Software

A good BASIC main program form is shown in Fig. 1.

```
REM
REM     Explain the overall program
REM
REM     What the first routine does
GOSUB
REM     Where we are now - e.g., what has been calculated, or input, etc.
REM     What the next subroutine does
GOSUB
REM     Where we are now
REM     What the next subroutine does
GOSUB
```

Fig. 1.

and so on . . .

A good BASIC subroutine form is shown in Fig. 2.

```
REM SUBROUTINE TO . . . . . (explain what it does)
REM IT WORKS BY . . . . . (explain how it does it)
REM IT NEEDS THE FOLLOWING VARIABLES . . . (list all inputs to the routine)
REM IT CHANGES THE FOLLOWING VARIABLES . . (all internal and returned names)
REM IT IS REQUIRED BY THESE SUBROUTINES . . (everything that calls it)
REM IT REQUIRES SUBROUTINES . . . . .  (everything that it calls)
then the rest of the subroutine.
```

Fig. 2.

Avoid using statement types that are local to the BASIC that you happen to be using. This means that you have to know what a few different BASIC dialects look like. In general DATA, DEF, DIM, END, FOR, GOTO, GOSUB, IF ... GOTO, INPUT, LET, NEXT, PRINT, READ, REM, RETURN, and STOP are included in every nonTiny BASIC. Functions that you can use safely are ABS, INT, RND (usually), SIN, SQR, and almost all full BASIC interpreters include COS, ATN, EXP, LOG, and TAN. All other statement types should be carefully explained so that the new users can implement the equivalent in their language. Segregate special statements into a few subroutines. They should never be part of the main program. If you write your BASIC programs as outlined here, your name will shine in the hearts of anyone who tries to use your programs. If not, you will be cursed in foul language known only to those frustrated by hours of trying to outguess you. *Programs should not be puzzles.*

The section on I/O also applies to BASIC. In other words, put all I/O in subroutines. GET and PUT, as well as START and FINISH, are often not appropriate in BASIC. But, if you use MITS' OUT function, for example, be sure to use a variable for the device number and similarly for the WAIT mask and control port number. If you are writing for a CRT terminal and you need a special byte to erase the screen, make the value of that byte a variable, and make a conspicuous REMARK explaining what it is doing. The next user can then either eliminate the erase with that strange byte, or change it to another code for his or her CRT. If you don't include the REMARK, then the next user won't know whether it is a special start-up code, rings the bell, or just patches around a bug in your system.

If a program has too many REMARKS to fit in your computer, *save the source listing with the remarks* and make a copy on paper or magnetic tape to run without the comments. When writing, create each program segment with full REMARKS, save the subroutine after testing it, and then, if you must to save space, eliminate the remarks from the copy in use. *Keep a fully documented source listing* and if you send your program to a fellow hobbyist (or to the *Kilobaud* Software Library), *send the fully documented version.* You may also send a compressed version, but that is not nearly as important.

Assembler Level Interchangeability Between MPUs

There is only one way: Excellent documentation (Where have you heard *that* before?). Extensive use of flowcharts helps, too. Another advantage is breaking down the program into small routines, each of which is easy to program from scratch. Some of the features needed in a good document are: Registers used, location and order of arguments, other stack usage, flags affected, *and anything else that could affect any other program that might become part of the system.* I won't go into a dissertation on the usual good practice of saving and restoring all registers, pointers, and flags whenever entering and leaving a subroutine. And then there is the problem of interrupts. Another day for that one.

Summary

Document like crazy. ■

Too many of us are still toggling programs and short routines in by hand. The time is long past for getting some sophistication into this operation . . . and Mike Aronson has a good introduction here to ASSEMBLIES . . . one way of doing just that. Once you begin using an assembler and the mnemonics (symbolic code) for the various instructions instead of octal or hexadecimal machine code, you'll wonder how you ever got along without one. Mike comes up with some very interesting points here, including some "don'ts", one of which is a program that modifies itself (bad news). He also defines a relocatable program *by showing us why and how one can and can't be (did anyone understand that?). His article really timely now that assemblers are becoming more and more available for hobby systems. — John.*

Welcome to Assembly Language Programming

Mike Aronson
156 Donald St
Oregon City OR 97045

Suppose a married couple went away for a week and left their teenage children in charge of the house. In most homes, a simple "Take care of the house while we are gone," would not be sufficient to keep the house on the lot for three days, let alone one week. So, the parents scheme in advance. Notes are prepared and left where their kids will find them.

"Put your dirty socks down the laundry chute," is attached to the record player. Just to make sure the job gets done, another note is taped to the inside of the hamper door.

"If you want to find the turntable spindle, wash the dirty dishes." Naturally, the spindle and another note are attached to the detergent.

"If you want to find the keys to the car, make your bed." As they do each job correctly, the reluctant housekeepers find new and explicit instructions which take them through the week. (But be careful if you try this type of reward system. Instead of a neat and tidy house, you might return to find your children have organized a treasure hunt for *you*!)

A new computer hobbyist, who has just finished building a microcomputer kit, has exactly the same problem as he or she connects up a terminal and turns it on. Suppose you decide the first thing you want to do with your beautiful, humming machine is to put eight zeros into every byte of memory. What do you do? You can't type "please clear memory" because the computer needs explicit instructions. You will have to write a series of statements that breaks the job up into small tasks that your machine can handle. This is called writing a program. There are, however, two little annoying problems: 1) The computer doesn't understand English, it speaks machine language. (Machine language is composed of ones and zeros — very boring.) 2) You can't be present while the computer follows your program. If you have made mistakes, the computer won't stop and ask questions. It will execute your program even though it is incorrect.

If you are a beginner, don't throw up your hands and give up. You can buy a computer program, that somebody else has written, which will allow you to talk to your machine. This is called an *assembler* program. Most assemblers are not very sophisticated. (That means they can't speak English either!) So you have to learn how to talk to an assembler program. Once you have mastered that, the assembler program will translate your programs into machine language for you. That solves the first problem. As for the second problem, well, nobody's perfect. My advice is, don't let the possibility of making some mistakes stand in the way of writing your own programs. This article will help you read and write simple programs for an 8080 assembler. Just remember, as you go on, there are three steps to computer programming: 1) decide the type and order of instructions; 2) assemble the program into machine language; 3) execute the machine language program.

First, let's take a look inside an 8080 chip (Fig. 1). (I've left out timing and some other functions that we don't need to worry about here.) Instructions and data come into the chip on an 8-bit parallel transmission line called the external data bus. These instructions and data are stored in memory locations outside of the chip. The microprocessor can pick up its instructions and data by

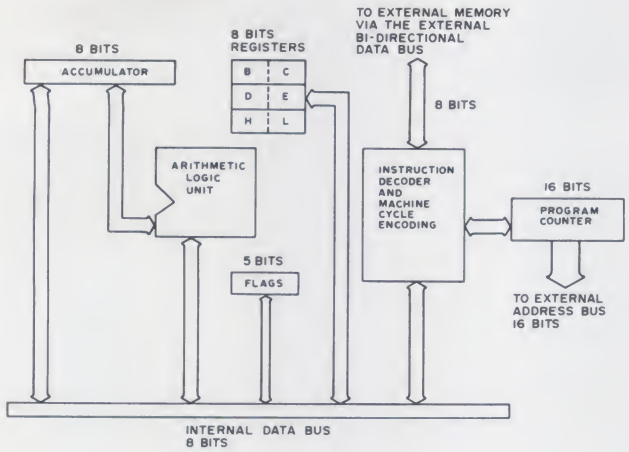


Fig. 1. Simplified 8080 block diagram.

selecting a 16-bit address (from either the program counter or the instruction decoder) and sending it out on the address bus. Whatever is in that memory location will appear on the data bus. The reverse is also possible. The chip can send data to memory locations by selecting an address and then presenting the data on the external data bus. The data will be stored at the selected location. The data bus is called bidirectional because data can flow in both directions, in or out of the 8080.

The internal data bus supplies communication between the instruction decoder and four other units. The arithmetic logic unit contains the rules for addition, subtraction, and how to perform logic operations. Six 8-bit registers are tied together into another unit of three pairs. They can be used as either 8- or 16-bit registers. The accumulator, a special 8-bit register used for arithmetic operations, can also be used for temporary data storage. There are five flip-flops called flags; we will only use three of them. One flag is set if the value of your last arithmetic operation is zero. For example, the result of a subtraction may be zero, or the contents of register C may be made zero by a subtraction. Whenever any arithmetic operation results in a value of zero, the zero flag is set. A second flag is set if the answer to the last arithmetic operation was a negative number. The third flag, the carry flag, is set if you attempt to put a binary number bigger than 8 bits into the accumulator. One example of this would be when two numbers are summed, causing an overflow. This flag is also set if a subtraction required "borrowing" from a place higher than the eighth bit. Flags retain the value they are set to until another arithmetic instruction changes them. They can be interrogated between arithmetic instructions and decisions may be based on their value. There are many examples of this later in the article.

Note, however, what is *not* inside an 8080 chip — there is no place to store a program. Program instructions and data are stored in the same place, external memory.

Now take a look at Table 1. I've selected some instructions that an assembler program recognizes. They are called data transfers because they move data from one register to another. The instruction decoder decides what goes where. Let's try to write our memory clearing program using these instructions. Suppose that we wanted to clear out one particular memory byte. Let's say the contents of memory at address 640 is to be zeroed. The microprocessor must put the 16-bit code that corresponds to 640 on the address bus and all zeros on the external data bus. But that is at execution time, step 3 of computer programming. We are still at step 1, deciding what instructions to use. So we jot this down on a piece of paper.

```
MVI   A
 0
STA
640
```

Each line represents a single instruction or data. An assembler program would take this series of instructions and numbers, in the order I wrote them, and change them into machine language code (i.e., 1s and 0s). If this machine language program were loaded into memory and executed, the first instruction would cause a zero to be placed into the accumulator (register A). The second instruction would store the contents of the accumulator in memory location 640.

This is not a bad first try at writing a program. The last byte, however, could cause some confusion. Do we mean decimal 640 or octal 640? Assembler programs need to be told what kind of numbers you are writing. I use a # when I'm expressing a decimal number and no symbol when I use octal numbers in programs.

There is another difficulty with the last instruction. STA instructions are supposed to be three bytes long. The reason for the extra two bytes is to have enough room to write a 16-bit address code. For an 8080 chip, the highest order bits are always put the furthest away from the instruction. For example, consider the decimal number, #640. In 16-bit binary it is written 0000001010000000. When converted to octal and two 8-bit values (i.e., divided in the middle), the *least* significant bits would be 200 and the 8 most significant bits would be 002 (00 000 010 10 000 000). If we rewrite our first program and remember to put the most significant bits of the octal numbers into the last byte, it would look like this.

```
MVI   A
000
STA
200
002
```

So this simple program is five bytes long. The assembler

would load this program into memory in five consecutive locations that are external to the processor chip. Once the program counter (see Fig. 1) is given the address of the first instruction of our program, the chip can take it from there. The instruction decoder will advance the program counter by the number given in the column called *Bytes* in Table 1. The address in the program counter, remember, is sent out on the address lines. Thus the decoder can skip over data that is mixed in with instructions. Only the first and third bytes, of the above program, contain instructions. The other bytes contain "operands." An operand is any additional information (data or address) needed by an instruction. One-byte instructions don't have operands. Two-byte instructions have one-byte operands. Three-bytes instructions have two-byte operands. If you are sitting near an 8080 type computer and have an assembler to load, try this program. Single step your computer to see how it works.

This is a good time to stop and consolidate what we have done so far.

Question: Suppose you turned on your computer for the first time, connected up a TV typewriter and typed

```
MVI  A
```

and pushed the carriage return. The letters

```
MVI  A
```

are readable on the TV screen. What was stored into the computer's memory?

Answer: Nothing. There is no program in memory that recognizes what you did. What you see on the screen comes from the terminal itself, not the computer. There is a good chance that turning on the power to your computer put random data and instructions into memory. Computer scientists have a technical word for this type of program — garbage. (And the chance that this garbage would write a machine language program that could accept your input is very small.) *You must have a program in memory to accept data from a keyboard if you want to communicate with your computer.*

Question: Suppose you purchase an assembler program and load it into computer memory. Can you communicate with your computer now?

Answer: Assuming you have bought a good assembler that is made for your machine and keyboard combination, you only have to do one more thing, execute the assembler program. Push the run/start switch. The assembler program is now running and waiting for input from the keyboard.

Question: Suppose you load and execute an assembler program. Why can't you type

```
PLEASE CLEAR MEMORY
```

and expect the contents of memory to be zeroed?

Answer: Intel decided what your 8080 chip could do when it designed the instruction decoder. As useful as this instruction might be to you, it isn't part of the instruction set. You must break the task up into instructions that the decoder can understand.

Question: Suppose you wanted the assembler program to write a machine language instruction that would cause the number in register D to be moved to register E. What would be the input to the assembler?

Answer:

```
MOV  E  D
```

Question: Suppose you typed

```
MVI  C
000
```

as input for an assembler program. After you are through, what is the content of the C register?

Answer: You don't know. All you have done is to store the machine language equivalent of

```
MVI  C
000
```

somewhere into memory. The instruction will be executed at a later time (i.e., the instruction is not executed while it is being "assembled").

Question: Suppose you type

```
MOV  A  B
MVI  B
001
LXI  H
020
000
```

as input to an assembler. How many bytes of memory will the machine language program occupy?

Answer: Consult Table 1 if you are not sure how many bytes each instruction occupies. The total number of bytes is six. Notice operands are mixed in with instructions. The instruction decoder will be able to sort it out.

Question: Suppose the previous program were loaded into memory and executed. After execution, what would be the contents of register L?

Answer: The LXI H instruction put a 020 into register L. Consult Table 1 if you don't remember why.

Memory refresh time is over. Now I'll show you how to write a PLEASE CLEAR MEMORY program.

A MVI instruction always takes up two bytes of memory and STA always takes

Mnemonic	Bytes	Execution time in usec Without memory reference	With memory reference	Instructions to the microprocessor during execution
MOV R_1,R_2	1	2.5	3.5	Take the contents of register 2 and *move* it to register 1. (For example, MOV H,B would transfer the contents of B into H.)
MVI R	2	3.5	5.0	*Immediately* after this instruction you will find one number. *Move* that number into register R.
LXI R	3	3.0	5.0	*Immediately* after this instruction you will find two numbers. *Load* the second number into register R. (For this instruction R is valid only for registers B, D, and H.) *Load* the first number into the register following R (see Fig. 1).
STA	3	6.5	—	*Store* the contents of the *accumulator* into memory. Use the address given in the next two bytes.
LDA	3	6.5	—	*Load* the *Accumulator* from memory. The data is located at the address given in the next two bytes.

Table 1. Short 8080 Instruction set: Data transfer instructions.

Mnemonic	Bytes	Execution Time in usec	Instructions to the microprocessor during execution
JZ	3	5.0	If the result of your previous arithmetic instruction was the number *zero*, follow Rule A. If the result was not zero, follow Rule B. **Rule A** **You must jump to the location specified by the next two bytes.** **Rule B** **Your next instruction will be found three bytes after this one.**
JNZ	3	5.0	If the result of your previous arithmetic instruction was *not zero*, follow Rule A. If the result was zero, follow Rule B.
JP	3	5.0	If the result of your previous arithmetic instruction was a *positive* number (0, +1, +2, +3, etc.), follow Rule A. If the result was a negative number (-1, -2, -3, etc.), follow Rule B.
JM	3	5.0	If the result of your previous arithmetic instruction was *negative*, follow Rule A. If the result was positive, follow Rule B.
JMP	3	5.0	Follow Rule A (no decision to make).
JC	3	5.0	If your previous arithmetic instruction tried to *carry* (or borrow) beyond bit 7, follow Rule A, otherwise follow Rule B.
JNC	3	5.0	If your previous arithmetic instruction did *not* try to *carry* (or borrow) beyond bit 7, follow Rule A, otherwise follow Rule B.

Table 2. Program branch instructions.

Mnemonic	Bytes	Execution time in usec Without memory reference	With memory reference	Instructions to the microprocessor during execution
INR R	1	2.5	5.0	*Increment* the number in register R by one.
DCR R	1	2.5	5.0	*Decrement* the number in register R by one.
INX R	1	2.5	—	Register R and the register *next* to R are to be treated as one 16-bit register. *Increment* that 16-bit number by 1. (B, D, and H are the only valid registers for this instruction.)
DCX R	1	2.5	—	Register R and the register *next* to R are to be treated as one 16-bit register. *Decrement* that 16-bit number by 1. (B, D, and H are the only valid registers for this instruction.)
ADD R	1	2.0	3.5	*Add* what you find in register R to what is now in the accumulator. Put the answer into the accumulator. Set the carry flag = 1 if there was a carry to a bit higher than seven (0-7). Set the carry flag = 0 if there was no high bit carry.
SUB R	1	2.0	3.5	*Subtract* what you find in register R from what is now in the accumulator. Put the answer into the accumulator. Set the carry flag = 1 if there was a borrow attempted from the eighth bit. Set the carry flag = 0 if there was no high order borrow attempted.
SBB R	1	2.0	3.5	*Subtract* what you find in register R (and the value of the carry flag) from what is now in the accumulator. (The carry flag = 1 might indicate a *borrow* from a previous subtraction.) Then finish as if this were a SUB R instruction.
HLT	1	—	—	There are no more instructions after this one. *Halt* your execution.

Table 3. Arithmetic instructions.

three bytes. In the interest of paper conservation and since the assembler program always knows how long each instruction is, let's simply write each instruction on one line from now on. If you need a "refresh" as to how long an instruction is, look at the column labeled "bytes" in the tables. Let's also assume our assembler program will ignore anything after a semicolon. Now, I can write comments about each instruction that will help you follow the logic of our programs on the same line as the instruction (or two, if the typesetter sees fit).

Suppose we want to clear out more than one byte. We could write Program A.

This would not be a very good solution if we really wanted to clear a large number of memory locations; since each instruction takes up more than one byte of memory, the program would be longer than the memory we wanted to clear! Branching instructions can solve this problem. They allow us to repeat the same instructions many times and to make compact programs. A branch in a program is like a fork in a road. The chip will select which path to continue on by using the condition of the flags mentioned earlier (zero, negative numbers, carry) to make its decision. Study Table 2 before you read on.

Branching instructions give a programmer flexibility because they can change the contents of the program counter. Suppose our clearing program starts in the first byte of external memory, that is, at address 000 000. See Program B.

When the third instruction is executed, the program counter will be changed to 000 000. This is called a loop, since the last instruction returns to an earlier instruction. Of course, this is not the world's most useful program. It will continue to repeat the same instructions until you push stop or pull the plug. We will have to write a program that stops itself after it goes through a loop a certain number of times. Please eyeball Table 3 for further instructions.

Suppose we want to go through the loop just five times. See Program C.

Again, we will load this program beginning at address zero. When execution is begun, a five will be placed into the C register. The second instruction begins at memory location 000 002. A zero is placed into the accumulator. The third instruction is at location 000 004. The zero is the accumulator is sent to address #640. The fourth instruction is at location 000 007. (Be sure to consult the tables if you forget how many bytes a certain type of instruction occupies.) The contents of register C is changed to four. The next instruction is at location 000 010. The previous arithmetic instruction had a result of four. Since four is not zero, the next instruction will be at location 000 004, the STA. (I'm writing for humans now, not for assembler programs. Low order bits are last.) Once again, the contents of the accumulator are sent out to memory, and register C is decreased by one. When the program arrives again at location 000 010, the C register has a three in it. Since three is not zero, the program jumps again to location 000 004. This continues until the C register is counted down to zero. This time, the zero flag is set, no jump occurs, and the program halts.

Now we know how to control loops. The register C is called a counter. We could have used any of the available registers for this program; there is nothing special about C. In fact, the 8080 chip can use registers as 8- or 16-bit counters that can count up or down, just by using the appropriate instructions.

However, we still have a difficulty with our program. Why would we want to zero out address #640 five times? We probably don't. Instead, let's clear out five consecutive addresses beginning at #640. See Program D.

Don't go past this paragraph unless you are certain you understand this program. The first address of this program is 000 000. A MVI instruction takes up two bytes of memory, so the address of the second instruction is 000 002. The third instruction is at address 000 004. The STA instruction is at location 000 006. What is at address 000 007? Answer: One of the two operands of the STA instruction, in fact, low order bits of the address we wish to clear. We put these low order bits into the accumulator, add one, and put them back into address 000 007. The next time through the loop, a zero will be sent to address #641. The address will be changed each time through the loop until register C counts down to zero and the program halts.

This program will work. We could zero up to #256 addresses this way. This limit is set because we used an eight bit register to count down. If we used a sixteen bit register, we could clear 64K of memory; but, since there is a possible serious difficulty with this method, please read on.

Suppose one day you notice that the first two instructions of your clear program can be replaced with

```
LXI   B,   005 001
;Put 1 into B and 5 into C
```

Fantastic. Not only are you saving one byte of memory, you are also saving 4 microseconds of execution time. So you make this change, load your new program into memory starting at address 000 000, and execute it. Everything seems ok. The program runs and stops. You check memory to make sure everything is working properly and discover only #640 is clear; the next four addresses still have data. What went wrong?

Since the beginning of the program occupies one less byte, every instruction from MVI A, 0 and down has moved up one address location. You forgot to change the first operands in the LDA and STA instructions from 007 to 006. Instead of changing the low order address bits, you were changing the high order bits!

Assembly programs have a feature that can save the day when such a problem is encountered. They will accept a name or label address instead of numbers. Program E lists our repaired program showing two examples of labels:

```
MVI   A,   000          ;Put a zero in the accumulator
STA    ,   200 002      ;Store the contents of A into #640
STA    ,   201 002      ;Store the contents of A into #641
STA    ,   202 002      ;Store the contents of A into #642
STA    ,   203 002      ;Ditto for address #643
STA    ,   204 002      ;etc.
```

Program A

```
MVI   A,   0            ;Put a zero in the accumulator
STA        200 002      ;Store the zero in address #640
JMP        000 000      ;Jump back to the beginning
```

Program B

```
MVI   C,   5            ;Put a five into register C
MVI   A,   0            ;Put a zero into the accumulator
STA        200 002      ;Store the contents of the accumulator into
                         address #640
DCR   C                 ;Decrease the number in register C by one
JNZ        004 000      ;If C is not zero, jump back to the STA instruction
HLT                     ;Stop if C is zero
```

Program C

```
MVI   B,   1            ;Put 1 into register B
MVI   C,   5            ;Set up C to count 5 times through the loop
MVI   A,   0            ;Put a zero into the accumulator
STA        200 002      ;#640 in octal
LDA        007 000      ;Low order bits first, for assembler programs
ADD   B                 ;Add one to what was in location 7
STA        007 000      ;Put the answer back into location 7
DCR   C                 ;Decrease the contents of register C by one
JNZ        004 000      ;If C is not zero, jump back to the third instruction
HLT                     ;Stop if C is zero
```

Program D

```
        LXI   B,    005 001     ;Put 1 into B and 5 into C
LOOP    MVI   A,    0           ;Call this address LOOP
        STA         PLACE       ;Store the accumulator in an address called PLACE
        LDA         006 000     ;Put low order bits of address PLACE into A
        ADD   B                 ;Add one
        STA         006 000     ;Change the operand of the 3rd instruction
        DCR   C                 ;Subtract one from C
        JNZ         LOOP        ;If C is not zero, jump to the address called LOOP
        HLT                     ;Stop after five times through program
PLACE   RMB   5                 ;What does RMB stand for?
```

Program E

When the assembler program reads the second line of this program, it knows that LOOP is not an instruction. It has a table of all the 8080 instructions and none of them are spelled LOOP. So, the assembler says to itself, "This must be a label. So far, I have used up three bytes of memory on this program. Since I started at address 000 000, the next address I can use is 000 003. From now on, whenever I see LOOP, I will translate it to mean 000 003 and remember to put the low order bits first." So, the assembler program will translate our jump correctly.

STA PLACE is another example of a label. This time, the label first appears in the program as an operand, instead of in front of an instruction, so it has to be treated differently by the assembler program. Addresses are only assigned when labels are encountered in front of instructions, not as operands. The address represented by the label PLACE is located immediately following the HLT instruction. The assembler is given this information by the pseudoinstruction RMB (Reserve Memory Bytes). RMB is called a pseudoinstruction because it is not used at execution time. It is only used by the assembly program to assign an address location to the label PLACE. In our program PLACE would be assigned to the address 000 024. The operand (5) tells the assembler how many address locations to reserve. So this program will clear the five memory locations behind the program itself.

There are still some serious objections to this program. *It is considered dangerous for a program to modify itself.* For example, this program could not be run twice without first resetting the STA operand back to the address represented by the label PLACE. Each time the program has finished execution, the STA operand has the value of PLACE + 5. In addition, this program must always be loaded into memory beginning at address 000 000 because of the operands of LDA and the second STA instructions. That means this program is not relocatable to another section of memory. Finally, STA and LDA are very slow operations. The total execution time for our program is 153 microseconds to clear five locations. We can do better.

I've given you a lot of information so far and we've still got a number of things to cover. Following are some more questions and answers to help you consolidate the previous material.

Question: A loop occurs when a branch instruction causes a program to return to a previous instruction. Can a branch instruction jump to an instruction that has not been executed?

Answer: Yes, a branch instruction can transfer control backwards or forwards in a program. As long as the conditions of the jump are satisfied, the instruction decoder will change the contents of the program counter. The contents of the program counter is the address of the next instruction.

Question: Suppose that at execution time, the contents of register B was 003. We could write it this way: B = 003. Also, suppose L = 002, H = 001, E = 001, and A = 002. What would be their contents after each of the following instructions was executed?

```
INR   B
DCX   H
ADD   E
SUB   A
```

Answer:

```
INR     B       B = 004, L = 002, H = 001, E = 001, A = 002
DCX     H       B = 004, L = 001, H = 001, E = 001, A = 002
ADD     E       B = 004, L = 001, H = 001, E = 001, A = 003
SUB     A       B = 004, L = 001, H = 001, E = 001, A = 000
```

Note that the last instruction can be used to clear the accumulator.

Question: Suppose you assembled, loaded into memory, and executed the following program:

```
        MVI   B,    007
        MVI   A,    001
AGAIN   ADD   A
        DCR   B
        JNZ         AGAIN
        HLT
```

How many times would the third instruction be executed? After this program reaches the HLT instruction, what would be the contents of register B and the accumulator?

Answer: Register B is being used as a counter. All of the instructions inside the loop that begin at the address labeled AGAIN will be executed seven times. After the HLT statement is reached A = #128, B = 0. Notice that ADD A will double any number that is in A as long as it is less than #127. Doubling any of the numbers from #128 to #255 will cause an overflow and not give a correct answer.

Question: Suppose that the following three consecutive instructions appear in the middle of a program you have written.

```
MVI   A,    000
JZ          PLACE
JMP         AHEAD
```

Will the address of the next instruction to be executed be PLACE or AHEAD?

Answer: You don't know. Jump instructions use the flags to determine if the jump should be made or not. Flags are set by arithmetic instructions only. The zero flag was given its present value by the previous arithmetic instruction. Since we don't know what the last arithmetic instruction was, we can't predict which jump will be made.

If we rewrite the program

```
MVI   A,    000
ADD   A
JZ          PLACE
JMP         AHEAD
```

The JMP instruction will never be executed. ADD A will set the zero flag and a jump to PLACE will follow.

Now we can jump ahead and write a better PLEASE CLEAR MEMORY program. In the 8080 chip, the H and L register can perform a special function. Instead of just using internal chip registers, we can access external memory by using the symbol M in place of a register symbol. The address of external memory that we wish to access must be in registers H and L. You probably noticed that Tables 1 and 3 had two different execution times for some instructions. Whenever external memory is referenced by H and L, execution time is increased.

We can write a short program that will attempt to clear all of memory. Let's assume our microcomputer has 64K of memory and our program starts in location 000 000.

Program F will attempt to clear itself out of memory! Eventually, a zero will be placed in address #9 and the program will be locked into a never ending loop. The operand of the JNZ instruction will be zero instead of 000 005. Each time through the loop, the contents of H and L will be reset to #65535; thus the program will never stop. In fact, the HLT instruction has been cleared out of memory!

```
        LXI   H,    #65535  ;Put the highest address in H and L
        MVI   B,    0       ;Put a zero in register B
LOOP    MOV   M     B       ;Send the contents of B to memory
        DCX   H             ;Decrease 16 bit register H and L
        JNZ         LOOP    ;If H and L is not zero, jump to Loop
        HLT                 ;Stop if H and L is zero
```

Program F

Program G

```
        LXI   H,   LOW      ;Address to start clearing memory to H and L
        LXI   D,   HIGH     ;Address to stop clearing memory to D and E
        MOV   A,   E        ;(Subtract the low
        SUB   L             ;address from the
        MOV   E    A        ;high address in two
        MOV   A,   D        ;subtractions.
        SSB   H             ;The difference is placed in
        MOV   D,   A        ;registers D and E)
        MVI   A,   0        ;Put a zero in the accumulator
LOOP    MOV   M,   A        ;Send contents of A to memory
        INX   H             ;Increase address for next time
        DCX   D             ;Decrease 16-bit registers D and E
        JP          LOOP    ;Zero is a positive number also!
        HLT                 ;Stop when D and E are negative numbers
HIGH    RMB         2       ;(2 locations each, are needed
LOW     RMB         2       ;to save 16-bit addresses)
```

What we need is a program, that can be specified just before execution time, to clear memory between any two locations so that we won't have to reassemble the program each time we change the locations to be cleared. The programmer can make sure the addresses to be cleared do not include the program itself. The following program is relocatable to anywhere in memory. It only requires that the start address be loaded into a location labeled LOW, and the stop address loaded into a location labeled HIGH before execution time.

Execution time for Program G is (13.5 x n) + 25 microseconds where n is the number of locations to be cleared. It occupies #30 bytes of memory because we must count the locations HIGH and LOW as part of our program. Those locations must not be cleared. The program is relocatable because we didn't specify any locations at all when we wrote the program. The assembler will make address assignments for us. The program doesn't even specify how many times we have to jump to LOOP. This is calculated each time we execute the program by finding the difference between our start and stop address. Notice we have also changed the type of branch from a JNZ to a JP instruction. This is because *the difference between our start and stop addresses is always one less than the number of bytes we want to clear.* If we stopped the program when the counter reached zero, we would not have cleared the last memory location. In this program, when the counter reaches zero, the program recycles one more time. When it returns to the JP instruction, registers D and E now contain negative numbers and the program halts.

Since addresses are 16 bits long, and the 8080 chip can only do 8 bit arithmetic, the subtraction must be done in two parts. The first subtraction will set the carry flag if register L is greater than register E. The second subtraction will continue the borrow if necessary, from the previous subtraction. If, just before execution time, the addresses that are stored in HIGH and LOW are accidentally reversed, registers D and E will contain a negative number after the second subtraction. One byte will be cleared, and the program will halt.

This article has introduced you to twenty 8080 instructions. With these, you can write a few simple programs. That should also be enough to give you the confidence to attack other instructions in the set and begin decoding other programs that you find in *KB* and other sources. I've included a couple of self-text questions (and answers) to reinforce your understanding of this article. Although, expert programmers will notice that some of the programs could have been written more compactly, beginning programmers will be happy to see, however, that I've only used the instructions introduced in this article.

Welcome to the assembly language programming world! ■

Test Yourself on the Above Article

If a problem gives you any trouble, load it into your computer and run it.

1.

```
        MVI   E,   5
        MVI   A,   0
BACK    ADD   E
        DCR   E
        JNZ        BACK
        HLT
```

a) How many times will the ADD E instruction be executed?

b) What is in the accumulator after execution?

c) How could you modify this program to put the sum of the numbers 1 through #10 into A?

2.

```
        MVI   E,   0
        MVI   B,   2
        LDA        NUMBER
        MOV   D    A
        ADD   E
BACK    JZ         STORE
        SUB   B
        JP         BACK
        MOV   A    D
        STA        ODD
        JMP        FINISH
STORE   MOV   A    D
        STA        EVEN
FINISH  HLT
NUMBER  RMB        1
EVEN    RMB        1
ODD     RMB        1
```

a) What does this program do? You have probably noticed how hard it is to follow a program that has no comments at all. Write comments for each instruction in this program.

b) What happens if "NUMBER" has a negative value in it?

c) Modify this program to store positive or negative numbers correctly. (*Hint:* Instead of subtracting 2 from negative numbers, add 2.)

Answers to Self-Test Problems

1.

a) 5

b) #15

c) change the first instruction to

```
MVI   E,   #10
```

Other solutions are possible. Try yours on your computer.

2.

a) If a positive number is placed in NUMBER, the program will determine if the number is odd or even and store it in the appropriate byte.

b) All negative numbers will be stored in ODD no matter whether they are odd or even.

```
        MVI   E,   0        ;Begin with a zero in register E
        MVI   B,   2        ;And a two in register B
        LDA        NUMBER   ;Put the unknown number into the accumulator
        ADD   E             ;Add zero to determine sign of number
        JM         MINUS    ;Jump is negative
BACK    JZ         STORE    ;Zero means you had an even number
        SUB   B             ;Subtract two
        JP         BACK     ;Jump if remainder is positive
DONE    MOV   A    0        ;(A negative number here means
        STA        ODD      ;your original number was odd)
        JMP        FINISH   ;You're done




MINUS   ADD   B             ;Add two to a negative number
        JZ         STORE    ;Zero means you had an even number
        JM         MINUS    ;Is the remainder still negative?
        JMP        DONE     ;If not, you started with an odd number
STORE   MOV   A    D        ;Store your original even number
        STA        EVEN     ;You're done
FINISH  HLT                 ;Halt execution
NUMBER  RMB        1
EVEN    RMB        1
ODD     RMB        1
```

Program H

Program H is presented for practice only. If you really want to know if a binary number is odd or even, just look at the least significant bit. ■

6 Digit LED Clock Kit - 12/24 hr.

$9.95 ea. QTY. 12 OR MORE — $10.95 ea. QTY. 6-11 — $11.95 ea. QTY. 1-5

KIT INCLUDES
- INSTRUCTIONS
- QUALITY COMPONENTS
- 50 or 60 Hz OPERATION
- 12 or 24 HR OPERATION

6-LED Readouts (FND-359 Red, com. cathode)
1-MM5314 Clock Chip (24 pin)
13-Transistors
3-Switches
6-Capacitors
5-Diodes
9-Resistors
24-Molex pins for IC socket

LARGE .4" DIGITS!
ORDER KIT #850-4
AN INCREDIBLE VALUE!

"Kit #850-4 will furnish a complete set of clock components as listed. The only additional items required are a 7-12 VAC transformer, a circuit board and a cabinet, if desired."

Printed Circuit Board for kit # 850-4 (etched & drilled fiberglass) $2.95
Mini-Brite Red LED's (for colon in clock display) pkg. of 5 1.00
Molded Plug Transformer 115/10 VAC (with cord) 2.50
NOTE: Entire Clock may be assembled on one PC Board or Board may be cut to remote display.
Kit # 850-4 will fit Plexiglas Cabinet II.



CABINET I
3"H,6¼"W,5½"D
CABINET II
2½"H,5"W,4"D
ANY SIZE/COLOR **$6.50** ea.

PLEXIGLAS CABINETS

Great for Clocks or any LED Digital project. Clear-Red Chassis serves as Bezel to increase contrast of digital displays.

Black, White or Clear Cover
2/$12.

RED OR GREY PLEXIGLAS FOR DIGITAL BEZELS
3"x6"x1/8" **95¢** ea. **4/$3**

MOBILE LED CLOCK

12 OR 24-HOUR OPERATION

12 VOLT AC or DC POWERED FOR FIXED OR MOBILE OPERATION.

SIX LARGE .4" DIGITS!

KIT OR ASSEMBLED — MODEL 2001



ACCURATE TIME WITH ADJUSTABLE XTAL TIME BASE

Approx. Size: 1¾"H x 4"W x 4½"D

BATTERY BACK-UP FOR POWER FAILURE OR TRANSPORTING FROM HOUSE TO CAR. ETC.

- 6 JUMBO .4" RED LED'S BEHIND RED FILTER LENS WITH CHROME RIM.
- SET TIME FROM FRONT VIA HIDDEN SWITCHES • 12/24-Hr. TIME FORMAT
- STYLISH CHARCOAL GRAY CASE OF MOLDED HIGH TEMP. PLASTIC
- BRIDGE POWER INPUT CIRCUITRY — TWO WIRE NO POLARITY HOOK-UP
- OPTIONAL CONNECTION TO BLANK DISPLAY (Use When Key Off In Car, Etc.)
- TOP QUALITY PC BOARDS & COMPONENTS - EXCELLENT INSTRUCTIONS
- MOUNTING BRACKET INCLUDED

KIT #2001 COMPLETE KIT (Less 9V. Battery) **29.95** EA. 3 OR MORE **$27.95** EA.
115 VAC Power Pack #AC-1 **$2.50** EA.

ASSEMBLED UNITS WIRED & TESTED ORDER #2001 WT (LESS 9V. BATTERY) **$39.95** EA. 3 OR MORE **$37.95** EA. Assembled Units May Be Mixed With Kits for Qty. Price
WIRED FOR 12 HR. OP. IF NOT SPECIFIED OTHERWISE

JUMBO RED LED's 12/$1.00 50/$3.95

6 Digit-LED Clock-Calendar-Alarm Kit

This is a complete, top of the line, Kit for the person that wants the best. Some of the many features and options are: 12/24 time, 28-30-31 day calendar, alternates time (8 sec) and date (2 sec) or can display time only and date on demand, 24 hr alarm - 10 minute snooze, alarm set indicator, 50/60 HZ. line operation or use with Xtal time base (#TB-1), built in OSC for battery back-up / AC failure, Aux. timer, CHOICE OF DIGITS.

Kit #7001B	6 - .4" Digits	$39.95
Kit #7001C	4 - .6" Digits & 2-." [Seconds]	$42.95
Kit #7001X	6 - .6" Digits	$45.95

Kits are complete (less cabinet) including PC boards, power supply, IC socket, 9 switches, 16 transistors and all parts required for above features and options **[All #7001 Kits Will Fit Cabinet I Above]**

60 HZ. XTAL TIME BASE KIT

Will enable Digital Clock or Clock-Cal. Kits to operate from 12VDC. Uses MM5369 and 3.58MHZ. XTAL. Req. 5-15VDC/2.5 MA. 1"x2" PC Board. Easy 3 wire hookup Accuracy: +- 2 PPM

#TB-1 [adjustable]
Complete kit $4.95 ea
Wired & Cal. $9.95 ea

PRINTED CIRCUIT BOARDS for CT-7001 Kits sold separately with assembly info. PC Boards are drilled Fiberglass, solder plated and screened with component layout. Specify for # 7001B or # 7001C (Set of 2) $7.95

JUMBO DIGIT CLOCK KIT

A complete Kit (less Cabinet) featuring: six .5" digits, MM5314 IC, 12/24 Hr. time, 50/60 HZ., Plug-Transformer, Line Cord, Switches, and all Parts.
[Ideal Fit In Cabinet II]
Kit #5314-5 **$19.95** ea. **2/$38.**

JUMBO DIGIT CONVERSION KIT

Convert small digit LED clock to large .5" displays. Kit includes 6-.5" LED's, Multiplex PC Board & easy hook-up info.

Kit #JD-1CC For common Cathode
Kit #JD-1CA For common Anode
$9.95 ea. 2/$19.

SCHOTTKY TTL

74S00	$.35
74S01	.40
74S04	.55
74S05	.60
74S09	.55
74S10	.40
74S15	.55
74S20	.50
74S22	.45
74S40	.45
74S50	.45
74S51	.55
74S60	.85
74S64	.55
74S73	1.25
74S74	.85
74S75	1.75
74S78	1.50
74S86	.95
74S107	.95
74S112	.95
74S113	1.40
74S114	.95
74S121	.95
74S133	.75
74S134	.75
74S138	1.75
74S139	1.50
74S151	1.95
74S153	1.95
74S155	1.95
74S156	1.95
74S157	1.80
74S158	2.50
74S174	2.50
74S175	2.50
74S181	2.95
74S182	1.95
74S251	2.75

DTL

930	$.09
932	.09
937	.09

LED DRIVERS

7447	$.95
7448	.95
75491	.65
75492	.65

VOLTAGE REGULATORS

LM 309 H TO-5	$.95
LM 309 K TO-3	1.25

7805	TAB	.95
7812	TAB	1.25
7815	TO-3	1.25
7815	TAB	1.25
78L 15	TO-5	.75
7824	TO-3	1.25
723	DIP	.75
723	TO-5	.75

DIGITAL CLOCK IC's

MM 5312	$ 4.95
MM 5314	3.95
MM 5375 AB	3.95
CT-7001	7.95
CT-7002	13.95
50380	3.95
MM 5369	2.50

XTAL

3.579545 MHZ. $ 1.95

PRESCALE

11C90DC	$15.95
95H90	9.95

DIODES

IN 4002	1A, 100 PIV	12/$1.00
IN 4005	1A, 600 PIV	11/$1.00
IN 4007	1A, 1000 PIV	10/$1.00
RECTIFIER	2.5A, 1000 PIV	4/$1.00
IN 914	SIL. SIGNAL	20/$1.00
IN 4148	SIL. SIGNAL	20/$1.00
DYAC	28V.	4/$1.00

PLUG TRANSFORMERS

12 VAC at 150 MA	$ 2.50
12 VAC at 500 MA	3.50

LINEAR

555	TIMER	2/$1.00
556	DUAL TIMER	.95
565	PLL	.95
566	FUNCTION GEN.	1.75
567	TONE DECODER	1.75

IC SOCKETS

PINS	1-24	25	100
8	$.25	$.22	$.20
14	.25	.22	.20
16	.28	.25	.23
18	.31	.28	.26
24	.50	.45	.40
28	.60	.55	.50
40	.75	.70	.65

MEMORY

21L02 1K RAM $1.95

EXAR

XR 2556	$ 1.75
XR 2567	$ 1.95

TRANSISTORS

2N2222A	TO-18	5/$1.00
2N3415	TO-92	5/$1.00
2N3704	TO-92	5/$1.00
2N4249	TO-92	5/$1.00
2N4400	TO-92	5/$1.00
2N4437	TO-92	5/$1.00
2N5089	TO-92	5/$1.00
2N5457 N	J-FET	.95

SWITCHES

ROCKER SPDT	6/$1.
MINI-SLIDE SPDT	5/$1.
REG. SLIDE DPDT	6/$1.
PUSH BUTTON N.O.	3/$1.
MINI TOGGLE SPDT	$1.30
MINI TOGGLE DPDT	1.50

TRANSISTOR SOCKET TO-5/18 GOLD PINS
5/$1.00

NYLON WIRE TIES
8" TIE-WRAP 100/$1.95
4" TIE-WRAP 100/$1.75

MOLEX PINS
REEL OF 1000 $ 8.50
STRIP OF 100 1.25

OP AMPS
3/$1.00

301	TO-5
709	DIP
709	TO 5
741	DIP
741	M-DIP
741	TO-5
747	DIP
748	DIP

DISCRETE LED's

JUMBO RED
12/$1.00
50/$3.95
100/$7.50

PC TRIM POTS
25K 6/$1.00
4.7K 6/$1.00

SPECTROL 10K 10 TURN
95c
4/$3.00

MISC. PRIME IC's

FAIRCHILD 9316 (74161)	$.95
75234 DUAL CORE SENSE AMP	1.50
MM 502H TO-5 SHIFT REG	.95

BUY 100 OR MORE IC's [Any Mix] TAKE 10% DISCOUNT.

Fairchild Super Digit FND-359



.4" Char. Ht. 7 segment LED RED Com. Cath. Direct pin replacement for popular FND-70.

95¢ ea, 10/$8.50
100/$79.00

SET OF 6 FND-359 WITH MULTIPLEX PC BOARD $6.95

25 AMP BRIDGE
$1.95 ea.
3/$5.00
100 PIV



TELEPHONE FORMAT KEYBOARD
BY Chomerics

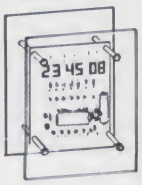


2-1/4"x3" 5/32" thick
$4.95
6/$28.

SEE THE WORKS Clock Kit Clear Plexiglas Stand

- 6Big .4" digits
- 12 or 24 hr. time
- 3 set switches (back)
- Plug transformer
- all parts included

Plexiglas is Pre-cut & drilled
Size: 6"H, 4½"W, 3"D



A SUPER LOOKING CLOCK!

Kit #850-4 CP **$23.50** ea. **2/$45.**

7-SEG LED

COMMON CATHODE

	COLOR	HT. DEC PT.	PR. EA.
FND-359	RED	.4" RHDP	$.95
FND-503	RED	.5" RHDP	$1.35
DL-750	RED	.6" LHDP	$2.95
XAN-654	GREEN	.6" NDP	$2.95
XAN-664	RED	.6" NDP	$2.95

COMMON ANODE

DL-747	RED	.6" LHDP	$2.95
XAN-72	RED	.3" LHDP	$1.25
XAN-81	YELLOW	.3" RHDP	$1.75
XAN-351	GREEN	.3" RHDP	$1.50
XAN-361	RED	.3" RHDP	$1.50
XAN-362	RED	.3" LHDP	$1.50
XAN-662	RED	.6" NDP	$2.50
XAN-692	RED	.6" NDP	$2.50

Form Inexpensive Sockets
100 for $1.25
Reel of 1000 - $8.50

MOLEX PINS



OPTOELECTRONICS, inc.

BOX 219 • HOLLYWOOD, FLA. 33022 • (305) 921-2056

master charge — BANKAMERICARD

ORDER BY PHONE OR MAIL
COD ORDERS WELCOME
[$1.00 CHG.]
Orders Under $15 Add $1.00 Handling
Fla. Res. Please Add 4% Sales Tax.

WE PAY ALL SHIPPING IN CONTINENTAL USA — OTHERS ADD 5% [10% FOR AIRMAIL]

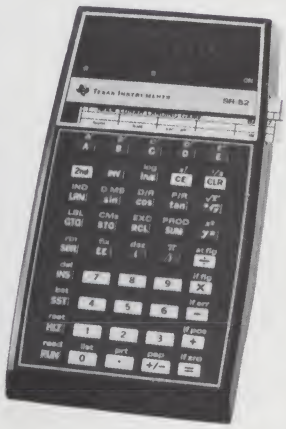
Over the years Pete Stark has written several books on computers and computer programming (one of which we offer through the Kilobaud *books section), plus a number of magazine articles on computers and digital devices. A long-standing contributor to* 73 Magazine, *Pete wrote an article on the construction of a digital counter several years ago which became one of the most popular construction projects in the entire country. In 1967 (which is* ancient *history in this game), Pete wrote an article for* Electronics World *in which he described interfacing a Morse code keyer to a PDP-8S computer he was teaching with. The following excerpt from that article reveals quite a bit about his insight and vision regarding the "home computer" (remember . . . 1967).*

"The price of $10,000 for a mere automatic keyer may seem somewhat steep, even considering its features. But remember that it can also do other things. In its spare time, the computer can balance your checkbook, do the kids' homework, or play tic-tac-toe with your neighbor. During a dull party it makes an excellent conversation piece; it can even be taught to tell your guests things you would never dare tell them yourself!"

"Programming? It's simple!" not only presents the beginning programmer with an interesting (and practical) method of getting into programming . . . it also has a couple of neat application programs at the end. If we can collect enough such programs, we will reprint them as a book. Everyone who contributes an accepted program will not only get paid for the write-up but also receive a free copy of the book. Send 'em in! — John.

Programming?

It's Simple!



Peter A. Stark
PO Box 209
Mt. Kisco NY 10549

In case you hadn't noticed, there's a revolution going in personal computing. Years ago only a very large corporation (or government) could afford a computer; now there are thousands of computer hobbyists who have their very own personal computers purring away happily in their basements.

Even more people are interested in these small computers but are scared away by the prospect of having to *program* them. When you build or buy a computer, that's only half the job. In order to use it, you have to give it programs — the instructions that tell it what to do and when. And the common feeling is that programming must be very hard — after all, professional programmers can get upwards of $20,000 a year for doing it, so it must be hard — right?

Wrong! Once you get the hang of it, programming is a cinch. And it's great fun. When I first learned to program a real computer, I spent nights at the office programming and running the computer just for the sheer fun of it — on my own time. Now that I teach programming, I find many students whom I literally have to push out the door of the computer room at night when we close. I may have trouble getting a half page of *home*work out of them, but put them into the computer room and they don't want to go home. It's addictive.

In this article, I will try to introduce you to programming in a new way. And best of all, I will show you how, for less than $80, you can get your own "programming machine" to practice on and see whether you like programming well enough to get your own full-scale computer system going. Once you get your feet wet, I bet you won't be able to resist going deeper into computing. What I am going to do is to describe how to program one of the new programmable calculators and show you how this is similar to programming a real computer.

Programmable calculators are available from several manufacturers, including Hewlett-Packard, Texas Instruments, Novus (National Semiconductor), and Sinclair. The ones made by H-P and TI are the best known of these The H-P models range in price from about $125 up to several thousand dollars. The models of interest to the average individual, however, are in the range of about $125 to about $400. These include the HP-25 ($125) which can run fairly nice programs, but as soon as it is

turned off it forgets the program you have entered; if you want to run it again, you must key the program in again. The HP-25C (about $175) is identical, but has a CMOS memory which remains powered even with the power switch off. As a result it remembers programs and data even when turned off. The HP-67 (about $400) is substantially more powerful and has the added capability of storing and saving programs on small magnetic cards. With a stack of magnetic cards you can build up a library of progams and feed them back at any time.

The TI models are roughly comparable. The SR-56 model (about $80) is about as complex as the HP-25, while the SR-52 (about $200) has a magnetic card like the HP-67. Novus, Sinclair, and others have several much less sophisticated models in the $80-and-under range. For a nonmathematical beginner, the TI units are simplest to use, so I will describe programming with them in mind While the $200 SR-52 is more powerful, can run longer programs, and has the magnetic card which greatly simplifies use, the SR-56 at $80 (at the time of writing — by the time you read this it may be even less expensive) is perfectly adequate to explore programming and we will use this as our example.

As we talk about programming this calculator, we will use some of the words normally reserved only for the big computers. In this way you will learn some computer lingo. Whenever we introduce a new word it will be *italicized, like this.* Keep in mind also, that all numerical values will be given in decimal (like 5, 19 or 78.325), rather than in the binary numbers normally used by computers. This will help you master the fundamentals of programming without getting bogged down in irrelevancies. So, here goes.

First of all, a programmable calculator has a display like that of a regular calculator. This display shows the numbers the calculator is using. A computer usually has a control panel on which there are rows of lights which also indicate the numbers being worked on The difference is that on the calculator the display is the only *output device,* whereas a computer may have additional output devices such as a printer or tape punch. The readout display on a computer's control panel is generally used only rarely, whereas on the calculator it is used all the time.

Internally, the display is connected to a display register. The word *register* generally refers to a group of flip-flops which actually holds the number, while the display only indicates the contents of that register. In computer lingo, this register would be called something like *accumulator, accumulator register, AC register,* or perhaps just *A register.* Simple arithmetic such as addition or subtraction is done in this register, which is frequently just a temporary stopping place for numbers on their way from one place in the computer to another.

In addition to the accumulator register, programmable calculators and computers often have other registers which are used for temporary storage of numbers. In some machines they can be used only for such storage, while on other machines they may be able to do very simple operations such as addition. These are called auxiliary registers and are labeled with numbers or letters. In the 8008 microprocessor, for example, these registers are labeled A, B, C, D, E, L and H. The SR-56 has ten such registers, labeled Register 0 through Register 9. (Notice how the numbers range from 0 through 9; not 1 through 10. This is common in computers, and it brings up an old joke — How do you tell whether a person is a programmer or not? Ask him to count to five. If he starts with zero, he is; if he starts with 1, he's not.)

Programmable calculators and computers also have a *memory*. This memory is used to store program instructions, data to be used in the program, as well as results of calculations. (This is the one big drawback to programmable calculators: Their memory can be used to store only programs and data to be worked on — never results. Results can go only into registers.)

On the SR-56 calculator the memory is divided into 100 separate locations, each with its own *address.* On the SR-56, these addresses range from 00 to 99 (Notice that again we start with 0, not with 1). If we use this memory only for program instructions, then we can put in up to 100 instructions; on the other hand, if we use part of it for data to be used in calculations, there will be room for fewer instructions. In most cases we are limited to much shorter programs both because part of the memory is needed for data (and results in real-life computers) and because some instructions require more than one memory location.

Each calculator memory location provides room for the storage of a simple number. In the case of the SR-56, each location can store a two-digit number such as 02 or 95. For example, memory location 00 might have stored in it the number 33, while location 01 might hold the number 21, and so on. The *contents* of a memory location is called a *word* in computerese. Since each word is two digits long, we say that the *word length* is two digits and that this is a *fixed word length* memory. In this case, we are measuring word length in decimal digits; in computers we use *bi*nary digi*ts* (or *bits*), and so we talk about 8-bit word lengths and so on. Obviously the greater the word length, the more information you can store in each memory location. The programmable calculator is hampered by its very short word length and thus results of calculations cannot be stored in their main memory — there is just not enough room! Only the display and auxiliary registers are long enough (13-digit word length) to store big numbers.

Let's now jump ahead a bit and consider an actual program. Suppose we have a number stored in Register 1 which we wish to add to another number stored in Register 3, with the answer to be left in the display register. An actual program to do this would go something like this:

```
Number in Register 1
Add
Number in Register 3
= (Leave answer in display register.)
STOP
```

Since this "program" is written in English, it now must be translated into keystrokes on the keyboard, so that it can be stored in the calculator's memory. After examining the keys on the keyboard and reading the instruction book, we find that: One key is labeled RCL (Recall — which means get a number from a register); while there is no key labeled Add, there is one with a + sign; and finally there is a key labeled R/S, which stands for RUN/STOP. Using these keys, the program becomes

```
RCL 1
+
RCL 3
=
R/S
```

This is a program written using symbols. In computer lingo, we would say that it *is* written in *symbolic code* (often called *assembly language*, because it is then used by something called an *assembler*). A program written in such a language is simply a description of what we want done, cut up into little chunks small enough for the computer (or calculator) to handle one at a time. We simply have to memorize which symbols the computer or calculator will accept (learn a new language), and

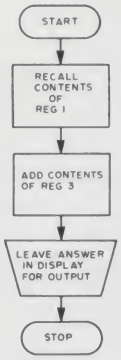


Fig. 1. Simple flowchart

then use them properly On a computer, all these symbols consist of letters (such as ADD, SUB, STOP, etc.), on calculators they may consist of letters or characters like + , - , or ÷ .

If you have been reading all this carefully so far, you probably have a question or two. We said a short time ago that this program must be stored in the calculator's memory. But we also said that each memory location can only store a two-digit number. How then do we store something like RCL 1 or " = "? The answer is simple — every time we press a key on the keyboard, the key generates a two-digit number. The RCL key is three keys down from the top and four keys in from the left, and so it is wired up to generate the number 34. The " = " key is in the ninth row from the top, four keys over,

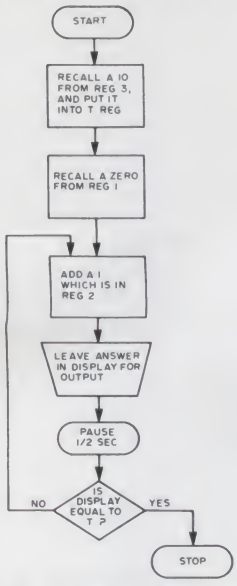


Fig. 3. Flowchart for counting to 10.

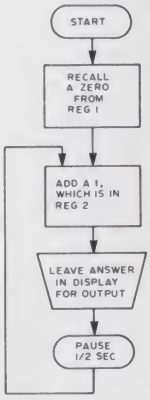


Fig. 2. Flowchart for counting 1, 2, 3, ...

and so it generates the code 94. The digit keys are treated a bit differently — the 1 translates into a code 01, and the 3 into 03. As soon as we press each key, the symbolic code gets translated into a numeric code:

Symbolic	Numeric
RCL 1	34 01
+	84
RCL 3	34 03
=	94
R/S	41

The numeric code, when used with computers, is actually called *machine language*. For each symbol in symbolic (assembly) language, there is a different numeric code. On a calculator, the translation is automatically done by the keyboard and associated circuits; in a computer the translation from assembly language to machine language is done by a program called an *assembler*, which uses a different procedure. Since different brands or models of calculators have different keys in different positions, they would have not only different symbolic languages but different machine languages as well The same applies to computers — learning to program one computer can teach us the basics of programming, but whenever we switch to a different model we have to look up the new codes (symbolic and machine) we need for that machine. When the very first computers were made, they were programmed entirely in machine language. Then, somewhere along the way, assemblers were invented and programmers switched to using assembly language. (The assembler is a fairly complex program which does the translation. Very small computers which do not have sufficient memory to run an assembler are therefore still programmed in machine language.) An even later development is a super-assembler called a *compiler* which can translate more useful and more easily learned languages, such as FORTRAN or BASIC, into machine language. But, since such a compiler uses even more memory than a ordinary assembler, you must already have a fairly large computer to be able to use it. Relatively few personal computers are big enough to run a FORTRAN or BASIC compiler. But enough of this detour — back to machine language.

We note that some instructions, like the R/S or stop instruction, were translated into a single number, like 41. This instruction is a very simple one and only consists of the *instruction* or *operation code* 41. On the other hand, the RCL 1 instruction was translated into 34 01. Obviously, the 34 stands for RCL, while the 01 specified which register. In this case the instruction consists of two parts — the operation code 34 and an *operand* 01. The operand specifies *where* or *to what* the operation applies.

If the calculator had a long enough word length, both parts of the RCL 1 instruction could be stored in the same memory location. In that case, each instruction would be stored in one memory location. Whenever a computer or calculator has a short word length, however, some instructions are too long to fit into one location. Depending on the type of instruction, we may have one word, two word, or even three word instructions mixed in the same program. This is one of the great differences between big computers and little ones — large computers tend to have long word lengths, and so relatively few words may be needed to make a reasonable program. Small computers on the other hand, usually have short word lengths, thus many words are required to produce the same program. For example, 8-bit microcomputers have 8-bit words, thus many of their instructions require two or three words. Likewise, the SR-56 has two and three word instructions, and so its 100-word memory limits us to programs of only a few dozen stops.

Let us now take the above program and actually put it into the SR-56 memory. Before putting it into memory, however, we must decide *where* in memory we shall put it. The usual convention is to pick a *starting address* and then put successive instructions into the following addresses. An easy choice is to start the program at location 00:

Memory Address	Memory Contents
00	34
01	01
02	84
03	34
04	03
05	94
06	41

A total of seven memory locations is needed for this program, out of the 100 locations available. (Note: The SR-52, 224-word memory permits longer programs.

Although the program's instructions are in some cases spread out over two (or even three) words, the calculator knows how many locations go with each operation code, and so it will perform the program properly. You may wonder why we needed the stop code 41 at the end of the program. Since the overall memory contains 100 locations, the other 93 unused locations might still have some other numbers in them from a previous use. If we did not put in the stop 41, the calculator would continue running past steps 05 and 06 into step 07, at which time it would probably start doing something else. It might never

stop, or it might output some really strange results.

Knowing how theprogram is stored, now allows us to go back and rewrite it in symbolic terms (see program A).

LOCATION	CODE	COMMENTS
00	RCL	Recall register 1.
01	1	
02	+	Add contents of register 3.
03	RCL	
04	3	
05	=	Leave the sum in display.
06	R/S	Stop.

Program (A)

This is the form in which we would usually write a program for the calculator. We would use the symbolic code and let the calculator do its own conversion. Note especially the comments column — when you return to the program a day or a month later, good notes help a lot to remind you of what you were doing.

Another device which helps you to document a program and even to design it in the first place, is a *flowchart*. This is simply a map of the program which describes the order in which things get done. Fig. 1 is the flowchart of this simple program. Note that the shapes of the blocks differ, depending on what instruction is being carried out. In this particular case, the flowchart is so simple that it is almost a waste of time to write it. This is seldom the case. Often, there are *loops* in a program — repetitions of certain parts of program over and over. Fig. 2 shows the flowchart of a program which flashes the numbers 1, 2, 3, 4, 5 . . . on the display. This program is a loop, which starts with a zero (which we have previously put into register 1), adds to it a 1 (which we have previously put into register 2), puts the answer (1) into the display, and pauses for a moment to let us read it. Then it goes back to add another 1. This program will repeat the loop until we turn the calculator off, each time increasing the number in the display by one and displaying the result. To write this program we need two new instruction codes: PAUSE, which causes the program to pause for about ½ second before continuing; and GOTO, which causes the program to go to a different memory location for its next stop. If you remember our previous program, we started the program at location 00 and placed each successive part of the program into the next location. Thus the calculator (or computer) would normally start at 00, go to 01, then 02, and so on. The GOTO instruction breaks that sequence causing the program to take its next instruction from a different location. A simplified program to do the job outlined in Fig. 2 would be something like:



Assigning each part of this to a memory location, we note that PAUSE is a one-word instruction, while GOTO is a 3 word instruction. The program as actually entered into the machine is program B.

LOCATION	CODE	COMMENTS
00	RCL	Get a 0 from register 1.
01	1	
02	+	Add a 1 from register 2.
03	RCL	
04	2	
05	=	Leave answer in display.
06	PAUSE	Pause ½ second.
07	GOTO	
08	0	Go back to step 02 to add again.
09	2	

Program (B)

See why the GOTO is a three-word instruction? The next two words give the place to GO TO — in this case location 02, which is the location of the add instruction.

The above program is a good example of a loop, but it has the disadvantage that there is no way to get out of the loop. It will continue until you manually stop the calculator. In most cases you will wish to provide a built-in stop. Loops are usually performed for a fixed number of repetitions or until they provide a desired result. This is done by means of *conditional GOTO* instructions, which do a GOTO only if a certain condition is satisfied. The conditional GOTO *tests* some internal condition each time the loop occurs. In the case of the SR-56, there are several test instructions we could use, along with an extra register called a T (or Test) register. Let us modify the flowchart of Fig. 2 to count only up to 10 (see Fig. 3). This chart is similar to that of Fig. 2, but with important differences — near the top, we initialize the T register to contain the number 10, and later we compare the contents of the display register with the T register. If they are the same we stop; if they are different we go back to add another 1.

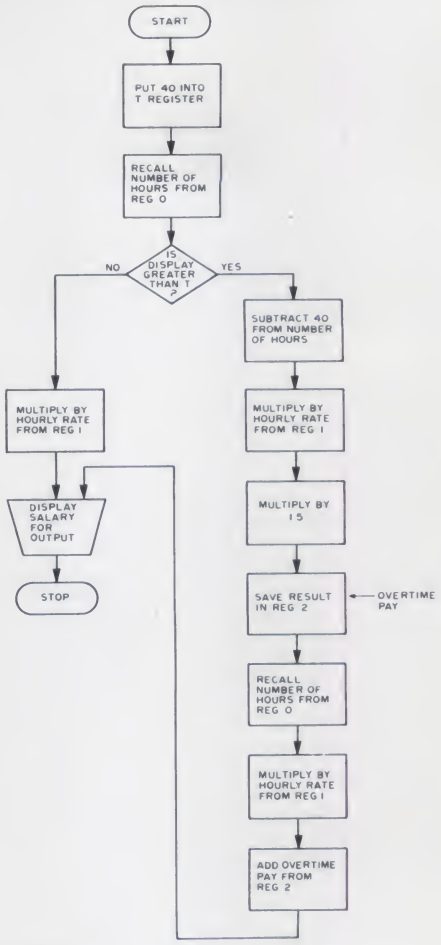


Fig. 4. Flowchart for computing weekly salary.

The flowchart is fairly easy to decipher if you start at the top and follow the arrows (making believe that you are the computer or calculator as you go along). Initially, we start with the display at 0, add 1, then pause with the number 1 showing. Then we check whether the 1 is the same as the 10 in the T register; since it isn't, we go back to add another 1, and next display a 2. We keep going around the loop like this until we display a 10, after which the diamond-shaped decision block shows us that we need to go to a stop.

In programming this problem, we need two more operation codes: X→T moves the number from the display register (called x on the SR-56) into the T register. "X=T?" checks whether they are the same. "X=T?" is actually a conditional GOTO

which does a GOTO if the condition is satisfied; it is a three-word instruction on the SR-56. (The complete program is shown in program C.)

shaped box) the program goes to the left and the result is simply the number of hours times the hourly rate.

If more than forty hours

Program (C)

LOCATION	CODE	COMMENTS
00	RCL	Get a 10 from register 3
01	3	
02	X-T	and put it into T register.
03	RCL	Get a 0 from register 1.
04	1	
05	+	Add a 1 from register 2.
06	RCL	
07	2	
08	=	Place answer in display.
09	PAUSE	Pause ½ second.
10	X=T?	
11	1	} If X=T GOTO location 16 to stop,
12	6	
13	GOTO	
14	0	} otherwise go back to step 05.
15	5	
16	R/S	Stop.

Conditional GOTOs can be used for jobs other than loops. For example, the conditional "X > T?" (is X greater than T?) would be used to make the decision in the overtime portion of the flowchart shown in Fig. 4. This program takes the number of hours a person works, together with the hourly rate, and computes the weekly salary (before deductions). If less than forty hours are worked (checked by the diamond are worked, then the answer at the test box is a YES. As a result, the program computes straight pay for the first forty hours and then adds time and a half for overtime pay for the hours over forty.

From some of the demonstration programs you have seen that really useful programs may take quite a few program steps. For this reason computers tend to have fairly sophisticated operation codes whose purpose is to do a lot with just one instruction. Each of the examples we have done so far could have been made shorter had we used some of the more complex instruction codes available. In addition to the simple operation codes we have touched on so far, the SR-56 includes many more. Some of these are more powerful than those of many large computers — for instance, the SR-56 can find square roots, sines, or logs in only one step, while actual computers generally need long programs to perform these same tasks. Thus even the 100-location memory of the SR-56 can be used to run quite useful programs.

For example, Fig. 5 shows a simple common-emitter amplifier. Given the values of the resistors and a few simple assumptions (such as that the transistor has a fairly good beta or that the emitter resistor is not too small), it is easy to calculate the voltages, currents and the gain. Fig. 6 shows the SR-56 program which does just that. Aside from a few new operation codes, such as STO which stores a number into an auxiliary register, this program demonstrates another computer trick.

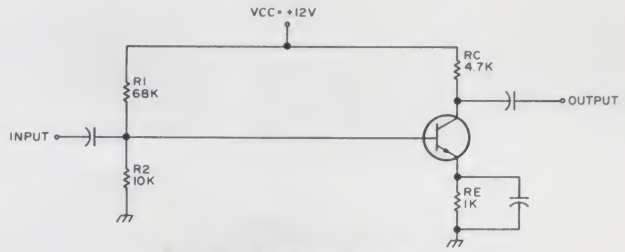


Fig. 5. Transistor amplifier.

Whenever a program needs a simple constant, such as 0.7 or 0.025, this constant could be stored in a register (or in a memory location of a large computer) and then recalled with an RCL instruction. But, if the constant is used only once in the entire program, this wastes a register which might be needed for something else. Instead, it is possible to put this constant directly into the program as shown in steps 16-17 or 42-45. In computers this is done by means of *immediate* instructions. For example, the 8008 microprocessor can load a number into the A register from memory by means of a LAM (load A from memory) instruction. But to do so requires that we also provide an address which goes with the instruction and tells the computer where to get the data to be loaded. It is much easier to use a LAI (load A immediate) instruction, and then put the number to be loaded into the very next location in memory after the LAI. We still have to put the number somewhere, but we avoid all the problems of addressing it. In a programmable calculator, this trick avoids the use of another register. ■

LOC	CODE	KEY	COMMENTS
00	34	RCL	Vcc
01	04	4	
02	64	X	X
03	34	RCL	R1
04	01	1	02
05	54		
06	52	(	
07	34	RCL	
08	00	0	(R1+R2)
09	84	+	
10	34	RCL	
11	01	1	
12	53	)	
13	94	=	=
14	41	R/S	Show VB
15	74	-	-
16	92	.	
17	07	7	.7
18	94	=	=
19	41	R/S	Show VE
20	54		
21	34	RCL	RE
22	03	3	
23	94	=	=
24	33	STO	Store

LOC	CODE	KEY	COMMENTS
25	05	5	
26	41	R/S	Show IE
27	64	X	X
28	34	RCL	
29	2	Rc	
30	93	+/-	Change
31	84	+	Sign
32	34	RCL	+
33	04	4	Vcc
34	94	=	=
35	41	R/S	Show Vc
36	34	RCL	Rc
37	02	2	
38	64	X	X
39	34	RCL	IE
40	05	5	
41	54		
42	92	.	
43	00	0	.025
44	02	2	
45	05	5	
46	94	=	=
47	41	R/S	Show Gain
48			
49			

REGISTERS	
0	R1
1	R2
2	RC
3	RE
4	Vcc
5	IE

NOTES	
Test case: if	
R1	= 68000
R2	= 10000
RC	= 4700
RE	= 1000
Vcc	= 12V
then	
VB	= 1.538V
VE	= .838V
IE	= .838ma
VC	= 8.059V
Gain	= 157.6

Program for finding base, emitter, and collector voltages, emitter current, and gain for the amplifier in Fig. 5.

Letters to the Editor

from page 67

I am sure that this one will be a thousand times better than the last. Although I still have my subscription to the original *Byte* and have gained from it, your new mag should offer even more for all us starters in the digital race.

I have enjoyed the I/O sections each month and will be looking forward to receiving KILOBAUD in the future. Thanks for having another go at it and helping the hobby now with your great publications.

By the way, the tear-out "bingo" card system this month is really handy. I have been wanting to mention it but have forgotten it every time I wrote. The one in *Byte* is tough to use and makes it hard to be sure that you are getting information wanted. I always have to look at its advertiser index, then look up the ad in the magazine. This takes a lot of extra time. On the other hand *Interface* puts its card numbers right under each ad. With this, I can make a quick pass through the mag when it arrives, check off any interesting items, and then go back through and study the rest of the material. Yours seems to be a cross between these two ideas and works pretty well, but you might take a closer look at something like that used by *Interface* (although this might be much more difficult to print). Anyway what you've done is a change for the better. Thanks again.

Pat Snyder WAØTTW
RR#3 Fremont NE 68025

EDP/Center — Desk or Phone?

I retired from my firm in 1969 and have been doing business counseling for international clients since then. Almost all of my clients were stalled with an EDP system they didn't understand or, what is most remarkable, refused to use. Generally the problem has been that the accounting department controlled the system and attempted to keep figures a secret, except for a very few executives.

In my own business (retail), I developed over the years a software system that used only a pencil and a small hand calculator. With this system (we grossed about $1 M per month), I was able to predict the final operating costs and net profit on a daily basis. Our monthly Profit & Loss came out ten days after close of the month. In the retail business this is too late!

I am interested in any new develop-

continued on page 116

CALL TOLL FREE 800-521-4414

NEW MOBILE DIGITAL CLOCK

AVAILABLE NOW –
- In-Dash
- On-Dash
- Black/Chrome
- Brown/Chrome

ENLARGED TO SHOW DETAIL



The DC101A digital clock sets the trend in digital automotive electronics which will be offered in the upcoming years. Its small size (1¼ x 2¼ x 5/8) and coordinated leather texture and chrome exterior make it an attractive addition to any car. Solid state electronics and a quartz crystal produce a clock that is the most modern timekeeping device for automobiles. It can be easily mounted in minutes in its compact ABS plastic case. Installation is accomplished by drilling an eighth inch hole in the dash, passing the power wires through it, and connecting them to the power with quick-connect wire clamps. Bright light emitting diodes (LED) allow for easy reading in all conditions. Since the clock is all solid state, it is inherently long life. A one year guarantee is offered on the clock.

- *Quartz Crystal Accuracy*
- *Light Emitting Diodes*
- *All Solid State Electronics*
- *Unit is Protected by Fusistor*
- *Black or Brown Case*
- *Installs in Minutes*
- *One Minute/Month Accuracy*
- *One Year Guarantee*

LIST PRICE ~~$59.00~~ INTRODUCTORY PRICE **$49.95**

Toll free U.S.A. 24 hour order & information line 800-521-4414. Outside U.S.A. & Michigan 24 hour phone 313-994-4441. Certified check or charge card on mail orders for immediate shipment. Dealer inquiries invited. Michigan residents add tax. Foreign orders invited. Call toll free or write for your free complete catalog & specifications. Satisfaction guaranteed or your money back. For engineering advice, call after 6:00 PM EST.

master charge THE INTERBANK CARD

COMMUNICATIONS ELECTRONICS™
P.O. Box 1002 Dept. 5
Ann Arbor, Michigan 48106

Your BANKAMERICARD welcome here

Structured Programming

... order out of chaos?

Okay. Bill Jones has provided us with some of the ground rules with which to carry on. His article on structured programming is of a straight tutorial nature, which is something we normally don't get too turned on about here at Kilobaud *(not that there's anything wrong with his article . . . it's just that we prefer to have a subject discussed using an example we can all relate to). Which brings me to my point: I would like very much for some of you software types who believe so strongly in structured programming (which is not to say I don't, for heaven's sake!) to get busy and run some comparisons. If we're going to "sell" the hobbyist programmer on these techniques then let's come up with some examples which clearly demonstrate the advantages of the new over the old. Then, write it all up into a fully-edited article, complete with diagrams, photos, and bells and whistles. Next step, of course, will be to send it to me so we can share it with the rest of the world (and make some bucks for you in the progress). — John.*

Bill Jones
541 Easy St
Marion OH 43302

Entry Point. Used to signify entry from another program. May contain the name of the particular program, routine or subroutine and always contains an entry point label.

Processing Block. Describes a function being performed within the logic flow.

Decision Block. Describes a function decision and indicates a two way branch.

Multiple Branch. Used as a functional branch where multiple decisions are possible or where multiple decisions are processed.

Connector. Denotes an entry point within a program segment or an entry from a different flowchart page. Contains a label used as an entry point or a number for continuation purposes.

Exit. Points to another program sequence or to another flowchart page. Contains an entry point label or a continuation number.

Label Block. Contains system and/or program defined labels when used in flowcharting.

Comment. Used when additional comments concerning a particular function are needed.

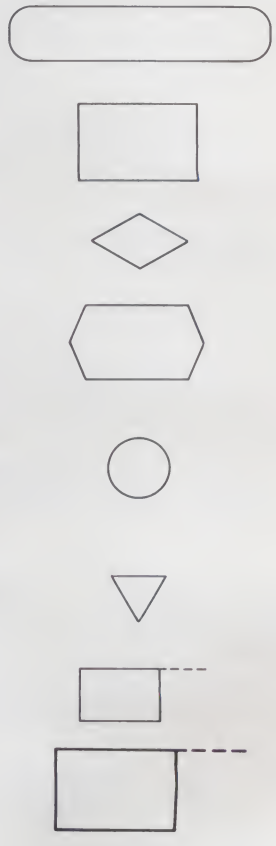


Fig. 1. Flowchart Symbols

In the past, design of software followed no particular design criteria which took reliability into account, and hence it was left to the individual to produce reliable code. While this has not been bad, it has not always produced reliable software. This is to say that the programs produced are not always correct. Many hours of testing go into an attempt to prove correctness after the design effort. The following guidelines are intended to ensure that any program designed using them will be *correct while* that program is still in the state of being designed and coded. Undoubtedly situations will come up in which it can be argued that the guidelines established are too restrictive, inefficient, or don't apply to the program under development; but at least these guidelines provide a common starting point which all can understand. (Needless to say, these guidelines apply to the personal systems programmer as well as any other.)

The program development design guidelines are taken from what is known in the literature as structured programming. Structured programming is based upon controlling the processes of program generation. These processes break program composition down into rules which are easily understood and establish "correctness" of design.

Control Structures

The control structures are given as the following sequence. The code for the flowchart symbols is shown in Fig. 1.

Simple Sequence. A simple sequence is one which performs a single function, operation or process. It makes no decisions. The flow of control is not relinquished within the operation. Control is given to the process and *must* return from the process. Mathematically it is linear reasoning. Symbolically, we represent this sequence as shown in Fig. 2.

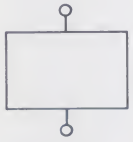


Fig. 2.

Decision Sequence. A decision sequence allows the result of a test to select the next executable statement. It is basically the IF-THEN-ELSE construct. While the decision is often a true or false answer it need not be. The decision could logically be a multiple decision tree. The flow of control is allowed to follow different paths within the sequence. However, when control is relinquished to the decision sequence, only one point of return from the processes within is allowed. Control is given to the decision sequence and then returned. The decision sequence is shown in Fig. 3.

Conditional Sequence. The conditional sequence allows the logical DO-WHILE construct. It allows several iterations of a process until a condition is met. Control from the sequence is not allowed to go on to the next executable statement until a decision is reached which allows it to continue. The conditional sequence is illustrated in Fig. 4.

General Conditional Sequence. An additional sequence, the general conditional sequence, is not much different than the conditional sequence but probably has more practical value. The sequence combines the conditional and simple sequence. This sequence also shows how one can *compose* sequences. Any *line of control* within a sequence can be *broken* into another sequence, thereby creating a nested set of sequences. The symbol we use to represent this sequence is shown in Fig. 5.

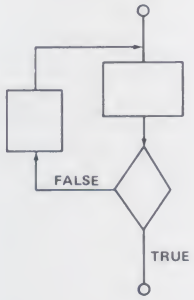


Fig. 5.

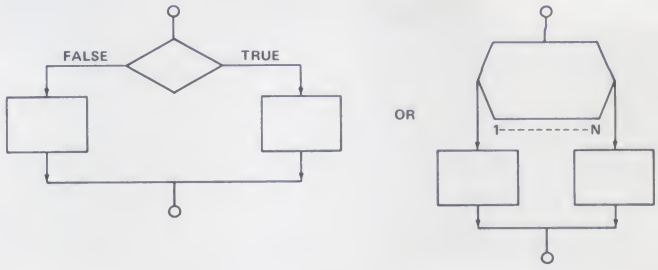


Fig. 3.

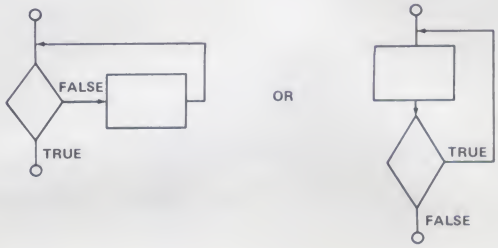


Fig. 4.

Single Entry/Exit. Each of the above sequences has the property of only one entry point and only one exit point. In essence each sequence decomposes into a 1-in/1-out *block box* which allows three types of control structures to develop the logic: simple (or linear), decision (or analysis), and conditional (or induction).

Slight alterations of the control structure diagrams are permissible. However, the control structure sequence must do two things: 1) perform logically the control intended and, 2) have a single entry/exit.

Program Flowcharts. All program flowcharts shall be realized by nested combinations of control structures. Any exceptions to this rule must be duly noted on the flowcharts.

To reiterate, the flowcharts are functional in content to reflect what is being accomplished at each step of the program. The flow diagrams are not intended to reflect the coding implementation but rather the logical functions which are to be per-

formed.

Flowcharts should start at the top of a page and progress downward. This gives the appearance of the control structures entry and exit points in logical order. (Also it is suggested that the flowcharts be designed on standard 8½x11 paper. The reasoning for this is to limit conceptually the size of the functional block or block box designed by the control sequences. This is not a rule but rather a guideline, as some functional blocks do require more space.)

Module. A module admits many descriptions. The term module will not be precisely defined but rather will be used to convey a concept. This will allow all design efforts to incorporate modularity. No matter what level of system description there is the concept of *dividing up into parts.* Each part is in essence a module. For example, a system is divided into input function, output analysis function, terminal scan function, etc. These parts of the system divide the system into modules. This unit then refers to a portion of the system. By defining how we divide the system, at the same time we specify the "connections" between the modules. Connectivity between the modules requires that the connection take place through the data base; i.e., the calling module does not imply information to the called module, the data is explicit.

When a module is further divided into submodules the same process of dividing should be used. While the connection of these modules may not be exactly the same, the assumptions made about the connections should be the same. Therefore, the assumptions which the modules make about one another have the same connection process.

While not always practical, the procedure for dividing modules (and thus determining the connectivity) should be the same at all levels. Regardless of the module, the connection between modules should contain as little information as possible and that information should be explicit. Since modules will not be defined at the same level of division within the system, the *same* procedure for dividing modules and definition for modules connectivity *must* always be used.

Module Documentation. Each module should have documentation in the listing. All module documentation shall use a double remark (**) to distinguish it from other remarks generated during the coding. The following information may be provided with each module.

1. Name
2. Functional Description
3. Input Requirements
4. Output Requirements
5. Other Modules Called or Used
6. Registers Changed
7. Special Requirements to Use the Module

Module Coding. Coding requirements must meet two criteria: efficiency and speed. While they are not necessarily compatible, they are essential. An additional requirement which cannot be measured objectively is comprehension. The module must be partitioned into meaningful and manageable units. These units are then in and of themselves modules.

Each module must have structure and format in order to comprehend and read the coding. For best management, code as few line statements as possible. Novels are nice but they take longer to read and comprehend.

Therefore to impose a discipline of writing programs a module should be limited to 50 or fewer lines of code. Comments per lines of code should be numerous and descriptive, relating to the function being performed rather than the implementation on the computer. ■

RS 232 INTERFACE CONNECTORS

25 CONTACT

	1-9	10-49	50-99
DB25S (socket)	3.90	3.25	3.00
DB25P (pin)	2.65	2.20	2.00
DB51226-1 (clamp)	1.60	1.30	1.00

We stock a full line in this family of connectors.

9, 15, 25, 37 and 50 contacts and coaxial units and a full stock of accessories, coax contacts hoods, shells, screw locks, sliding locks, etc.

—We have "data phone" types—

ORDERING INFORMATION:

California residents add 6% sales tax

All orders prepaid

Foreign orders — U.S. funds only

Orders under $15.00 add $1.00 shipping and handling

Prepaid orders over $15.00. We will pay freight

A-OK ELECTRONICS, INC.
5855 W. Centinela Ave.
LOS ANGELES, CA 90045
(213) 670-2266
TELEX 6533438

THE COMPUTER MART OF NEW JERSEY

501 Route #27
Iselin NJ 08830
(201) 283-0600

WANT SATISFACTION? We give expert advice and instruction so you can choose the RIGHT hardware and KNOW how to use it.
TRY US, YOU'LL LIKE US!

WE STOCK
*IMSAI
*Lear Siegler
*Tarbell
*Seals
*TDL
*Cromemco
*Processor Tech.
*Polymorphic
*Icom and more

WE SUPPLY
*Service
*Education
*Advice
*Software
·Apple
·Zapple
·Disk Basic
·Macro-Assembler and more

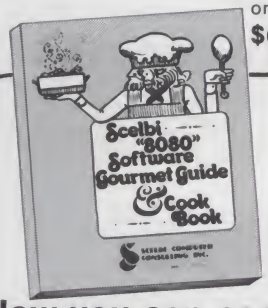
Send SASE for your free 8080 reference card.

master charge THE INTERBANK CARD

Your BANKAMERICARD welcome here

STORE HOURS:
Mon. — Sat. 10 AM to 6 PM
Evenings:
Mon. & Thurs. Open until 9 PM

only $9.95 ppd



Now you can cook-up hot programs on your "8080"

A gourmet's delight of practical "how to" facts, including description of "8080" instruction set. How to manipulate "8080" stack. Flow charts. Source listings. Routines for multiple precision operation. Programming time delays for real time applications. Random number generators. Completely assembled floating point math program. Input/output processing for basic I/O programming through interrupt processing. Code, numeric conversion routines. Real time programming. Search/sort routines. Plus many more finger-lickin' goodies.

Order your copy of **Scelbi's "8080" Software Gourmet Guide & Cook Book** today! Only $9.95 ppd. Bon appetite!

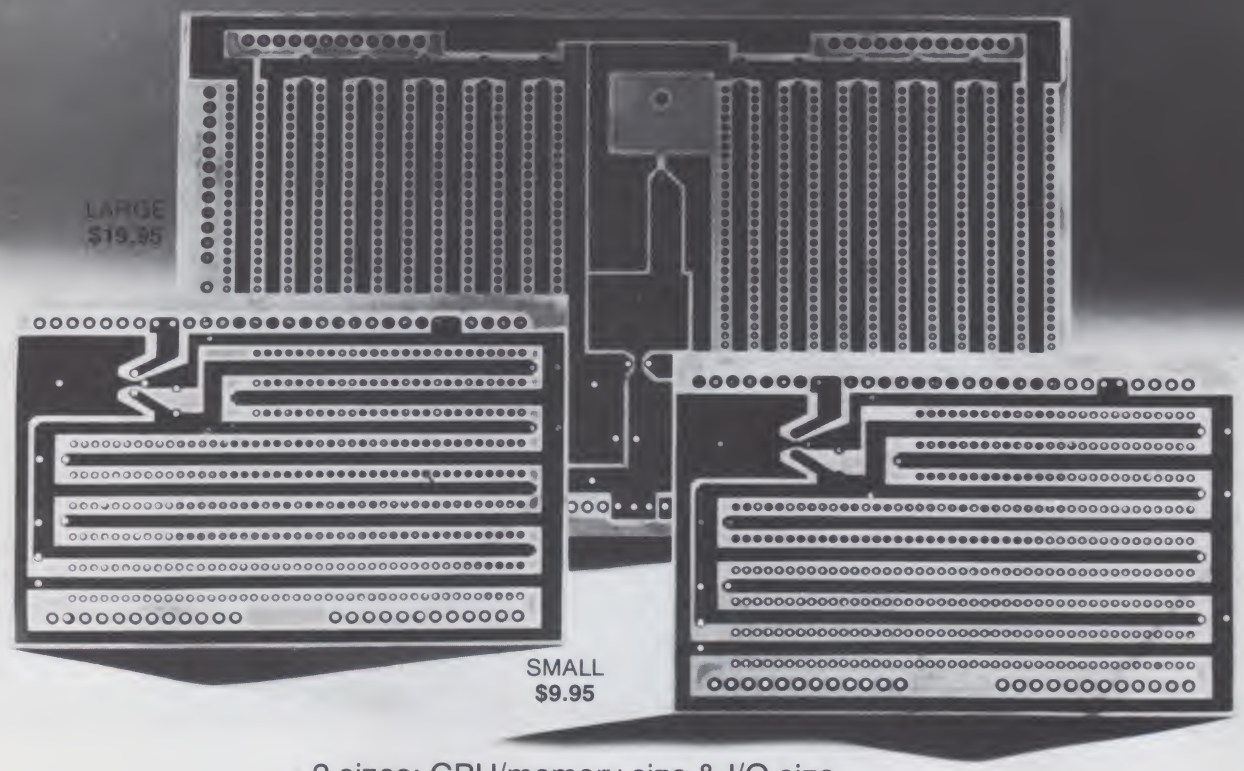
SCELBI COMPUTER CONSULTING INC.

1322 Rear Boston Post Road
Milford, CT 06460 • (203) 874-1573

6800?

Now!! Have One Your Way

The 6800 system owner can now have his best ideas in hardware on a buss compatable card designed to mate the SWTPC 6800 system.



- 2 sizes: CPU/memory size & I/O size
- Will accept 14, 16, 24 and 40 pin connectors
- Test and/or interface connections on top
- 2 on-board regulator locations (1 on small board)
- Short, low inductance power and ground
- Use with wire wrap
- Use with wiring pencil

SEND MONEY ORDER, CHECK OR BANK AMERICARD # (We prefer Bank Americard)

Personal Computing Company
3321 Towerwood Drive, Suite 107
Dallas, Texas 75234

DEALERS INVITED

Computers in Golf

... help for the handicapped

```
CLOAD D

OK
LIST-12Ø

1Ø REM TO SCRATCH TYPE NINE COMMAS AND THEN CR
2Ø REM TO INSERT HANDICAP, TYPE 2ØØ THEN NINE COMMAS THEN CR
3Ø REM THEN TYPE IN HANDICAP AND THEN CR
4Ø REM TO CORRECT AN ERROR, TYPE Ø IN THE LAST ENTRY
45 PRINT"TYPE DATE":INPUT A$
5Ø PRINT :PRINT
6Ø PRINT TAB(28)"HOUND EARS CLUB"
7Ø PRINT
8Ø PRINT"   MEN'S HANDICAP - COURSE RATING 68.2          DATE: ";A$
9Ø PRINT:PRINT
1ØØ AE$="ADDY, E.":AF$="ALDEN, J.":AG$="ALEXANDER, V."
11Ø AH$="ANDERSON, A.":AI$="ANDERSON, F.":AJ$="ANDRE, M."
12Ø BA$="BEAN, B.":BB$="BEAN, L.":BC$="BEAN, P."

5ØØ DEFINT A
51Ø DIM A(2Ø)
52Ø GOTO 8ØØ
53Ø PRINT "MISTAKE IN ENTRIES"
54Ø INPUT A(1),A(2),A(3),A(4),A(5),A(6),A(7),A(8),A(9),A(1Ø)
55Ø IF A(1)=Ø THEN E=Ø: IF A(1)=Ø THEN GOTO 72Ø
56Ø IF A(1)= 2ØØ THEN GOTO 74Ø
57Ø INPUT A(11),A(12),A(13),A(14),A(15),A(16),A(17),A(18),A(19),A(2Ø)
58Ø IF A(2Ø)=Ø THEN GOTO 53Ø
59Ø F=Ø
6ØØ FOR I=1 TO 19
61Ø IF A(I) =A(I+1) THEN 64Ø
62Ø SWAP A(I),A(I+1)
63Ø F=1
64Ø NEXT I
65Ø IF F=1 THEN 59Ø
66Ø PRINT "TEN LOWEST SCORES ARE ";
67Ø PRINT A(1);A(2);A(3);A(4);A(5);A(6);A(7);A(8);A(9);A(1Ø)
68Ø T=A(1)+A(2)+A(3)+A(4)+A(5)+A(6)+A(7)+A(8)+A(9)+A(1Ø)
69Ø B=T/1Ø:C=B-68.2:D=C*.96:E=INT(D+.5)
7ØØ IF E 4Ø THEN E=4Ø
71Ø PRINT "HANDICAP IS ";E
72Ø PRINT
73Ø RETURN
74Ø INPUT E:GOTO 71Ø
8ØØ PRINT AE$:GOSUB 54Ø:REM      ADDY
81Ø AE=E:PRINT AF$:GOSUB 54Ø:REM ALDEN
82Ø AF=E:PRINT AG$:GOSUB 54Ø:REM ALEXANDER
83Ø AG=E:PRINT AH$:GOSUB 54Ø:REM ANDERSON,A

185Ø PRINT"ADJUST PAPER AND CONTINUE"
186Ø STOP
2ØØØ PRINT:PRINT:PRINT
2Ø1Ø PRINT TAB(28)"HOUND EARS CLUB"
2Ø2Ø PRINT
2Ø3Ø PRINT"   MEN'S HANDICAPS - COURSE RATING 68.2        DATE: ";A$
2Ø4Ø PRINT:PRINT:PRINT
21ØØ PRINT AE$;TAB(15)AE;TAB(25)GA$;TAB(4Ø)GA;TAB(5Ø)PA$;TAB(65)PA
211Ø PRINT AF$;TAB(15)AF;TAB(25)GB$;TAB(4Ø)GB;TAB(5Ø)PB$;TAB(65)PB
212Ø PRINT AG$;TAB(15)AG;TAB(25)GC$;TAB(4Ø)GC
213Ø PRINT AH$;TAB(15)AH
214Ø PRINT AI$;TAB(15)AI;TAB(5Ø)RA$;TAB(65)RA
215Ø PRINT AJ$;TAB(15)AJ;TAB(25)HA$;TAB(4Ø)HA;TAB(5Ø)RB$;TAB(65)RB
216Ø PRINT TAB(25)HC$;TAB(4Ø)HC;TAB(5Ø)RC$;TAB(65)RC
217Ø PRINT TAB(25)HD$;TAB(4Ø)HD;TAB(5Ø)RD$;TAB(65)RD
OK
```

Fig. 1. The golf handicapping program (in Altair Extended BASIC). Rather than reproducing the entire data base of member's names, only a few are included as examples.

When I was involved with the Micro-8 Newsletter, as a co-editor, one of the most interesting letters I ever published was written by George Haller describing his golf handicapping program. (At the time I didn't consider the program very practical *... but definitely interesting!) George's original program was written for his Scelbi 8008 system; however, this article, reflects his update of that program for his Altair 8800/Disc system using MITS Extended BASIC. I have also been updated in that I can see, and appreciate, the small business money-making potential here. We can also thank George for coming up with a suggestion for a "BASIC Questions & Answers" section in* Kilobaud *which led to the* BASIC Forum *by Dick Whipple and John Arnold. — John.*

George L. Haller
1500 Galleon Dr.
Naples FL 33940

Handicapping for golf associations is not only a most interesting application for a small computer but can also be a possible source of revenue.

Calculating The Handicap

Club handicaps are usually calculated on a monthly basis and are done in a variety of ways. If the group has few members the golf professional usually does the calculations. As the club grows, a handicap committee is formed which meets once a month and does the work, probably using a hand calculator and allocating parts of the job to various members of the committee. If the size of the group increases to much over 100 members, the job of handicapping becomes too onerous and relief from outside services is sought. If the association is quite large, say over 400 members, then a full computer service with a storage terminal at the club is quite often used. The business applications for the small computer operator is usually

found in the medium sized club and fees from $2.00 to $3.00 per member per season can be charged. The club professional, or a member of the handicap committee, would be the first person to talk to about the service.

Depending on the size of the group, the handicapping is done in different ways, but always using the same basic formula. This formula takes the average of the ten best scores from the most recent twenty games played. From this average the United States Golf Association course rating is subtracted, and the result is multiplied by 0.96 and rounded. This result is then the individual player's handicap. For example, if the average of the ten best scores out of the most recent twenty games of regulation 18 holes is 93.2 strokes, and the course rating is 70.1, then the handicap is integer 22 (93.2 - 70.1) (0.96 = 22). The handicaps are used to match players of varying degrees of skill. All holes on a golf course are rated according to their difficulty from 1 to 18, with 1 being the most difficult hole. In a handicap match players are allowed to deduct strokes from their hole scores according to their handicap. Thus, a player with a 22 handicap would be allowed to deduct (from his or her hole scores) one stroke on each hole and an additional stroke for each of the holes handicapped at 1 through 4. A player with a 9 handicap would only be allowed to deduct one stroke each on handicap holes 1 through 9.

Handicaps can also be determined by taking the difference between two players' handicaps and using that value in the following manner: The player with a 22 handicap playing against the player with a 9 handicap would be allowed a stroke on each of the handicap holes 1 through 13 (the difference between 22 and 9), and the player with the 9 handicap would play "scratch" (no strokes deducted on any holes). In the case of a foursome, the player with the lowest handicap would play from "scratch" and the other three players would be allowed strokes based on their handicaps. The professional players that we see playing on TV are all considered to be zero handicap or "scratch" players. Handicapping doesn't enter into TV or professional tournaments, but most amateur play is done on a handicap basis and the handicap is a very important part of each player's record.

The Program

The program (Fig. 1) using ALTAIR Extended BASIC, is one used at a club that has a total of about 175 men and women players. Running separate programs for both men and women takes about four hours per month. Rather than listing the entire program, we've given excerpts which show the pertinent parts of the program without long lists of repetitious entries. Basically, the program prints a member's name and then accepts the 20 most recent scores. The scores are then sorted and the ten lowest scores are printed. The handicap is then calculated and printed, and the next member's name is printed for the inputting of his scores. This material is turned over to the club professional for member reference. Next, part of the program prints a summary of the names and scores which is then posted on the club bulletin board (Fig. 2).

Now . . . let's take a look at the details of the program. Lines 10 through 40 refresh the operator's memory as to how to handle the inputs. Lines 20 and 30 explain how to handle a direct handicap input if no new scores have been posted and line 45 inputs the date to be used in the headings. Lines 100 to 499 are used to list the members' names with their two letter string variable.

```
                    HOUND EARS CLUB

   MEN'S HANDICAPS - COURSE RATING 68.2          DATE: 1 AUGUST 1976

ADDY, E.          18    GARLAND, C.       11    PEARSON, E.       17
ALDEN, J.         11    GOODMAN, E.        6    PHILLIPS, C.      17
ALEXANDER, V.     10    GRYDER, D.        16
ANDERSON, A.      12                               RANKIN, J.        32
ANDERSON, F.      15    HALLER, G.        23    RIDENHOUR, J.     14
ANDRE, M.          8    HARTLINE, H.      11    ROBBINS, H.       11
                        HENSLEY, S.       16    ROBBINS, S.       10
BEAN, P.          14    HERMANN, R.       24
BIRL, W.          18    HUNTLEY, H.       11    SCATTERGOOD, A.   28
BOWNESS, R.       17                            SCHLESINGER, R.    9
BRADY, C.         11                            SCHLOSS, J.       11
BRADY, E.         17    JOHNSON, J.        7    SCOTT, W.         18
BRANSTROM, W.     12    JOHNSON, R.       10    SHAPARD, R.        7
BREWER, C.        19                            SHEALY, J.        16
BUTLER, W.         9    KEISTER, E.        7    SHOWELL, L.       31
                        KEYES, C.         15    SINGER, L.         9
CAUSEY, G.         5    KILLEBREW, C.     34    SMITH, R.         12
CHEEK, L.         14    KOSTEN, L.         6    STEPHENS, P.       8
CHURCH, J.         0    KRAMER, E.        14    STOKES, C.        18
CLARK, W.         12                            SUTHERLAND, J.    12
CLOGSTON, R.       8
COFFEY, F.        15    LAFFERTY, M.      26
COLEMAN, F.       15    LEATH, W.          8    TATE, S.          16
COLVERT, H.       16    LEFCOURT, S.      20    THOMAS, J.        13
CONE, B.          28    LISK, B.           6    THOMPSON, G.      17
COX, A.           17    LUCE, C.          13    TINGLE, C.        27
CROTHERS, H.      14                            TUCKER, L.         9
CUSHING, A.       20    MC KAUGHN, R.     13
                        MC LAURY, E.      21    VELDE, L.          9
DAVIS, F.         14    MABRY, C.         24
DILLING, J.        7    MANN, J.          10
DURANT, R.        17    MARION, J.        10    WALSWORTH, C.     15
                        MICHNOFF, D.       9    WAUD, O.           8
EDWARDS, W.       11    MITCHELL, N.      18    WILLIAMS, J.       7
ELFMON, S.        33    MORRIS, L.        18    WILLIAMS, W.      23
EWING, M.         19    MORRISON, S.      10    WISE, F.          14
                        MORRISON, R.       6    WOLFSON, R.       14
FALLOWS, R.        4    MOTES, E.          9    WOOTEN, R.        12
FERRIS, F.        25
FICHTENGER, C.    22    NEWMAN, H.        19
FISCHLEY, W.      30    NEWTON, E.         7
FURR, F.          13
FURRY, G.         18

OK
```

Fig. 2. Print-out of club member's handicaps.

Lines 500 and 510 set the dimensional array. We then go to line 800 which prints the first member's name and goes to a subroutine starting on line 540. Line 540 calls for an input of the first ten scores. Line 550 scratches that entry if the first entry is a zero. If the first entry is a 200, then the subroutine is terminated and the numeric handicap (variable "E" in the program) can be inputted directly.

Fig. 3 shows a heading and an example of a normal input of 20 scores under the name "Addy, E." An example of a directly inserted handicap is shown under the name "Alden, J." Line 570 inputs the second 10 of the 20 scores. These scores were divided into two blocks of ten for terminal width reasons. If a mistake has been made in the entry of any score, we enter zero for the last score and the computer

```
OK
RUN TYPE DATE
? 15 AUG 1976

                    HOUND EARS CLUB

   MEN'S HANDICAP - COURSE RATING 68.2          DATE: 15 AUG 1976


ADDY, E.
? 87,86,88,87,85,86,94,88,89,91
? 85,89,89,90,90,89,86,92,88,88
TEN LOWEST SCORES ARE  85  85  86  86  86  87  87  88  88  88
HANDICAP IS  18

ALDEN, J.
? 200,,,,,,,,
? 11
HANDICAP IS  11

ALEXANDER, V.
?

OK
GOTO2000
```

Fig. 3. Sample print-out during updating or entering of new scores for computing of handicap.

UP YOUR.....

Memory

Fast, Low Power 2102

$1.60 Beat This!!!

4K RAM Board Kit $79.95

- LOW POWER 2102
- DENSE 4½"X 6" PACKAGE
- STANDARD 44 PIN GOLD PLATED CONNECTOR

6800 OEM Prototype Board

INCLUDES SOCKETS $42.50

For orders under $25.00, add $2.00 shipping and handling

WASATCH SEMICONDUCTOR PRODUCTS
25 South 300 East, Suite 215
Salt Lake City, Utah 84111

OEM INQUIRIES INVITED

prints "MISTAKE IN ENTRIES." We then start over with that player's entries. Lines 590 through 650 are a sorting routine. If your BASIC does not have a "SWAP" command, another sorting method will have to be used. Lines 660-670 print the ten lowest scores. Lines 680 through 700 calculates the handicap and assigns its value to the variable E. The handicap is then printed, with the statement "HANDICAP IS," under the ten lowest scores. We then return to the next player name input, line 810. Note that the first statement in line 810 stores the previously calculated handicap value E in the double letter variable AE, which is associated with the player's name string variable $AE.

This operation continues until we have completed the inputting of scores for all members. We then arrive at lines 1850-1860 which allows us to adjust the paper, so that we can start on a complete sheet, before giving the "CONT" command. The computer then prints out a summary sheet as shown in Fig. 2. More copies of this sheet can be made by adjusting the paper and giving the command "GOTO 2000."

Conclusion

This program, like most, could probably be improved. A suggested next step would be to save all scores as data and input only the new scores, which would delete the older scores and still leave a total of twenty. Another would be to print the old handicap as well as the new on the summary sheet. This would be a much more complicated program, but would save some time in inputting scores where fewer than twenty scores were posted for the past interval. There are probably other ways to change the program, but it is running successfully as described here. That is, of course, the main objective. ■

6502 OWNERS

assembler/text-editor program
In source and object form.

ANSI basic program

star trek game program

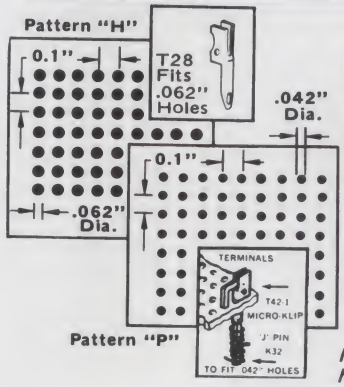
All above programs run in 4K memory. Available on either cassette or paper tape.

PROGRAMS $19.95 EACH PPD.
(Texas Residents add 5% sales tax)

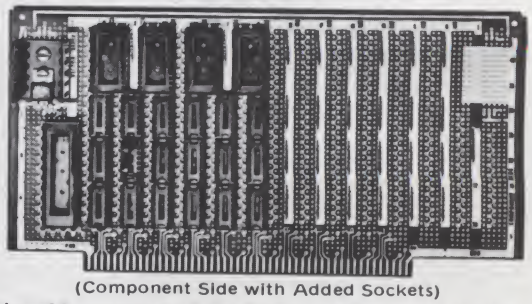
Hard copy available with purchase for an additional $12.95 ppd. Membership in User Group with purchase. Complete listing of programs available on request.

MICRO SOFTWARE SPECIALISTS, INC.
2024 Washington Street
Commerce TX 75428
(214) 886-6300

VECTORBORD® TENTH/TENTH HOLES MOUNT:
DIPS, SIPS, KLIPS, CHIPS
PINS, POSTS, POTS, PADS **Save Work — Time — Money**
RCs, ICs, PCs, SCs



8800V MICROPROCESSOR PLUGBORD



(Component Side with Added Sockets)

Has 100 contacts on 0.1" centers, is 10" wide by 5.313" high. Has heavy tinned back-to-back buses, overall 0.1" spaced 0.042" hole pattern. Socketed models available.

WIDE SELECTION OF SIZES AND MATERIALS

MICRO-VECTORBORD® "P" — 0.042" holes match DIP leads. Epoxy glass, or glass composite, paper, copper clad. Also 1/64" to 1/16" thick and 10" max. width.

VECTORBORD "H" — For larger terminals, leads. Available in epoxy glass sheets 4.8" to 8.5" wide and 8.5" to 17" long. 1/32" and 1/16" thick.

TERMINALS — Complete selection of wire wrappable and solderable push-in terminals for 0.042" and 0.062" dia. holes — plus wiring tools available.

PLUGBORDS — For solder or wrap wire construction 2.73" to 10" wide and 4.5" to 9,6" long. With holes .1"x .1", .1"x .2", .2"x .2", or loaded with IC sockets.

Send for complete literature
Vector ELECTRONIC COMPANY, INC.
12460 Gladstone Ave., Sylmar CA 91342
(213) 365-9661 — TWX (910) 496-1539

41576

After you get through splitting a gut on this one, be sure and show it to your wife. She'll enjoy it for sure. — John.



Computer Widow

Barbara Henderson
3998 Berrywood Drive
Santa Maria CA 93454

Do you think *you* have problems? I'm married to one of the *worst* computer nuts the world has ever known. And when my husband becomes interested in a hobby, we all better look out! His dedication in building his microcomputer system this past year has been nothing short of a trial for me.

Now that I think about it, I can see that he has been preparing me for this heavy involvement for quite a few years. First came the radio-controlled model airplanes, with everything from a scale model J-3 to a homebuilt with a nine-foot wing span. Presently there are seven airplane carcasses hanging in the garage, all in various states of disrepair. After the model airplanes, a private pilot's license was next on his list. However, after a few short (and harrowing) flights and a couple of white-knuckle landings, he conceded that this hobby was not very practical. Then, there's the water-ski boat out beside the house; it's been in the water *once* in the past four years. And, the complete welding set-up — just in case anything ever needs to be welded. Oh, and we mustn't forget the motorcycle which he rides maybe twice a year.

But, his interest in these past hobbies appears to be *nothing* compared to his obsession with computers. I think we have enough computer equipment around our house to be able to replace IBM at a moment's notice. On the larger side, there are the two eight-foot tall strip chart recorders which were too good a deal to pass up. Then, there are the five paper tape punches (does one computer system *really* need that many?). As far as keyboards go, we are now the proud owners of four, with the explanation that each one was just a little better than the others and he was really trying to get one that *I* would enjoy using. That sounds like quite a bit of equipment for one Altair 8800, doesn't it? But the best is yet to come. We added on a larger family room, to have additional space to live in (I thought), but our old family room has now been taken over by two *gigantic* disc drives. Since they each cost about $8000 new, he got such a good deal, I'm sure it would have been criminal to pass them up! When friends come to visit, they wonder if the discs are my new model washer and dryer, or his-and-her steam baths, or just what. The disc drives can't be out in the garage workshop with all the other equipment, though. Why? Because they can't be around all those soap and bleach fumes from the clothes washer! I sometimes wish that I were as pampered and well cared for as his equipment. I'll bet there's a lot more 24 karat gold in that Altair than there is in my jewelry box!

I guess it should be understandable that all this equipment would require his seemingly constant attention, but there are more things being neglected around our house than just me. Our children hardly recognize their own father unless he's perched on his work stool in the garage. Then there is the backyard fence which was blown over last spring in a heavy wind. The new family room has a lot of unfinished electrical work and wood paneling that hasn't been touched for eight months. There are leaky faucets in the kitchen and bathroom which are rusting from neglect. And get this: There is a rather conspicuous 300 gallon hot water tank beside the house, waiting to be used some day in a solar heating system. Maybe. But in the meantime I'm about to run out of Excedrin tablets!

I've been thinking about forming a Central California Computer Widows' Club for all of us neglected wives. I'm sure there are many out there who feel as I do (as a matter of fact, it should probably be a *national* organization). We could get together and learn how to program and repair computers so we could spend more time with our husbands, or maybe learn how to sabotage the monsters so we could get even! Or maybe we could just get together and cry a lot.

After due consideration, though, I don't think I would trade my husband for anybody, even though he appears to have terminal computer disease. At least he doesn't spend every night in front of the TV, sacked out on the couch, or out with the boys. And, whenever I need him, all I have to do is open the garage door . . . and there he is. So I can talk to you any time I want, can't I, Honey? Honey? Are you listening? Honey? ■



The Digital Group System (posing for a photo). Starting at top-center and going clockwise: the Sanyo Monitor; substitute El Cheapo (but good) Roberts cassette recorder (normally a Superscope C-104); some of the gobs of software available on cassettes; the keyboard; the motherboard, with the four boards of a basic system plugged in (8K of RAM, CPU board w/2K of RAM, video and cassette interface board, and the I/O board with four input and four output ports) with room for memory and I/O expansion; and last, a more than adequate power supply.

What's that Digital Group Really Doing?

This article was written some months back, and since then the Digital Group has come up with a nice collection of cabinets to house the various subassemblies which make up their system. With a couple of additions, the comments and observations I made regarding their system still hold true (90% favorable, I might add). The first additional comment concerns the Digital Group monitor program and the fact it resides in page zero. This seems to cause problems when it comes to loading other software in almost every instance. One owner, I know even went so far as to put a switch in which allowed him to select between the monitor's PROM or RAM memory in page zero. Naturally, he still isn't able to use the PROM for loading programs into page zero . . . the answer seems to lie in relocating the monitor elsewhere in memory. The second additional comment concerns the fact the Digital Group didn't go with the Altair bus. Granted their bus layout may even be better than the Altair, there are still a lot of occasions when a Digital Group owner would like to have one of those peripherals, or options, which plugs into the Altair. On the other hand, there are probably a lot of Altair system owners that would like to be able to use the Suding Phi-deck controller or video driver or whatever. If someone has developed a Digital-Group-to-Altair interface (or is thinking about it), Kilobaud Magazine *would be tickled pink to pay you well for the plans (i.e., a construction article). — John.*

John Craig
Editor

If you can design and build a microcomputer system that will play the Star-Spangled Banner through an AM radio and at the same time generate an American flag on a TV screen, it doesn't really *have* to do anything else! The average layman will be so fascinated by this demonstration that the last thing in the world he'll ever ask is, "But what are you going to use it for?" (You've heard that one before, right?)

Dr. Robert Suding WØLMD is the man behind the design of the Digital Group System, and you can rest assured that the system is capable of doing much more than playing the Star-Spangled Banner. One of the main objectives behind the design of the system was to make it a truly "turn-key" system (i.e., build it, connect the various components, apply power, and begin operating). I'm here to tell you he/they did it!

There are several things about the system which are impressive just from first glances. Quality prevails in almost every instance (and, I'll be sure and clue you in on the one or two places I feel it doesn't . . . keep count, I think it's only ONE). Upon walking up to one of these systems, your eyes first fall upon the 9 inch Sanyo monitor (not a regular TV) and the sharp, clear characters being displayed. The character set includes both upper and lower case, as well as the Greek alphabet (just what you always wanted, right?). The quality of the PC boards is equal to, if not better than, boards I've seen put out by some of the biggies. The cassette recorder sold with the system is a Superscope C-104 with variable speed and other neat features. It's a rather expensive unit ($119.50), but then it's really the heart of the system and they figured it shouldn't be compromised. I might add, however, that the system I worked with didn't have this recorder with it, so I proceeded to connect my $24.95 Roberts recorder (pictured in the upper right-hand corner of the photo) and didn't have any problems with reading prerecorded programs.

The keyboard was the one item which I didn't get too excited about. In the documentation supplied with the system they make quite a point of the fact that if you're dissatisfied with any item they welcome its return. Therefore, if a body felt as I did about this keyboard, it could certainly be returned for a refund or an exchange, and I understand they will be offering a different model (or models) in the future.

CPU-Independent Architecture and Upgradability

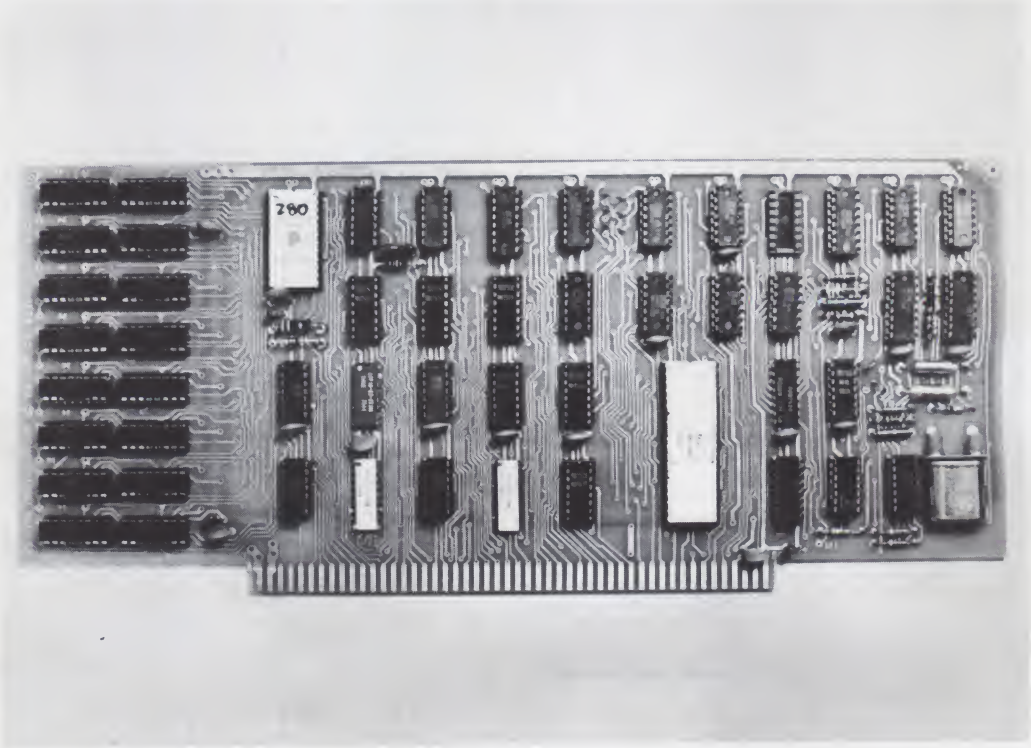
A very unique, and certainly worthwhile feature of the system is the fact that the architecture isn't dedicated to any one particular microprocessor. The 8080 CPU board is the most popular, but DG also offers a CPU board which has your choice of a Motorola 6800 or a MOS Technology 6502 (with only two or three jumpers necessary to go from one to the other). There's going to be a lot of software developed within the hobbyist community for all three of these processors, and there aren't too many systems around offering this capability.

And, talk about upgradability!! How's about upgrading to the ZILOG Z-80?? The Z-80 has got to be one of the hottest items to come out of the Silicon Valley in quite some time . . . and DG has put together a CPU board using the little beast. Here are just a few of its worthy features: 1) Refresh for dynamic memory is generated internally within the chip, and the refresh address register can be accessed for *random number generation*! 2) Two sets of six general-purpose registers, which can be combined in pairs to form 16-bit registers. 3) Instructions for moving any size block of memory and also for searching a block of memory for a particular value. 4) Bit manipulation instructions. 5) And more . . .

Construction

The unit I worked with was already built and running, but a friend of mine, Robert White, recently constructed a DG system and had some worthwhile comments. Perhaps the most significant was the fact he felt the construction documentation was adequate for the job. He brought up the system by plugging each board into someone else's unit (which is one of the advantages of finding another unit in your area . . . during the firing up of *any* system). He also mentioned that a scope was necessary for checking the timing and performing the cassette board alignment. On a scale of 1 to 10, he gave the unit a resounding 10.

The Digital Group System doesn't have a front panel. If you've ever had occasion to troubleshoot a computer



The new Z-80 CPU board (for only $50 more than the 8080 board!).

Features (displayed on the screen after reading in the TBX-TVCOS cassette):

1. READ BASIC Program (from cassette)
2. WRITE BASIC Program (to cassette)
3. Display Commands (displays an unannotated list of TBX commands on TV screen)
4. Display Error Codes (annotated list of TBX error codes listed on TV)
5. Continue Programming (useful for exiting and re-entering BASIC program during debugging)
6. TINY BASIC (causes entry into TBX for writing BASIC programs)

TBX Commands:

LST – List Program
RUN – Execute (run) resident program
NEW – Start new program (& "scratch" old)
SZE – Display resident program size
PR – PRINT output data
IN – Enter (INPUT) a value from keyboard
LET – Assign a value to a variable
IF – Conditional test statement
FOR – Beginning of a loop
NXT – End of loop
GOTO – Alter program execution sequence (Jump)
GOSUB – Jump to subroutine
RET – Return from subroutine
DTA – Assign value(s) to simple or single dimensioned array variable (equivalent to LET)
DIM – Assign array dimension(s)
END – End of program

Table 1. Tiny BASIC Extended TV-Casette Operating System, Features and Commands.

without a front panel, perhaps you can appreciate the value of one. (Very handy for single-stepping through a program or entering short troubleshooting routines . . . so you know exactly what the machine is doing, rather than trying to also figure out what's going on with the Monitor program.) The Digital Group folks apparently didn't go the front panel route because of the reliability of their unit. But for those desiring one, they have very thoughtfully provided construction plans for building one.

The Software!

Digital Group Software Systems, Inc. (DGSS) is a company formed by DG for software distribution. The operation is run by Charles and Dianne Howerton in Arvada, Colorado. They try to ship all orders within a 24 to 48 hour time frame . . . and if they can't, notify the customer. All shipments are guaranteed, and they've replaced several cassettes that have developed "glitches" after being shipped. They pointed out that it is very unwise to place cassettes on top of, or beside, TV monitors . . . especially those with metal cabinets.

When power is initially applied to a Digital Group System, the message "READ 8080 INITIALIZE CASSETTE" appears on the monitor screen. One simply starts the cassette recorder at the beginning of the program to be loaded, and when a steady tone begins, the reset button is pushed and released. At the end of the tone leader, the program is read into main memory. The TV displays the least significant digit of the octal page currently being loaded, byte by byte (so you can "see" the program as it's coming in). Memory is also checked byte by byte and missing or defective memory addresses are indicated by a period (".") rather than the octal digit of the page. This cassette loader program in ROM is further evidence of the "turn-key" characteristic of the system.

Following is a list and brief description of some of the neat software available from Digital Group Software Systems:

TINY BASIC EXTENDED TV-CASSETTE OPERATING SYSTEM – Developed by Dick Whipple and John Arnold of Tyler, Texas. Tiny BASIC Extended (TBX) was their baby, and Dr. Suding interfaced it with TV and cassette drivers to come up with the Operating System (TBX-TVCOS). A very neat package, and easy to use. Table 1 lists the features and commands available with TBX-TVCOS.

8080 OPERATING SYSTEM (standard with the system) – Provides the user with cassette read and write routines, memory dump, and machine language programming capability from the keyboard. Also has a "calling" feature so that the user can call up a particular program or routine in main memory and run it. This cassette also includes five demonstration and diagnostic programs:

1. *Computerized Amateur Radio* . . . with a "CW Keyboard" routine (speed selectable) and a "RTTY Receive" routine for converting frequency shifted 60 wpm Baudot to ASCII and displaying the characters. (Very simple to implement.)
2. *Synthesized Music* . . . described earlier . . . that's the program that plays the Star-Spangled Banner while drawing an American flag on the monitor. (Interesting insofar as a regular AM radio is used to pick up the music . . . resulting from frequency modulating the data paths with different timing loops.)
3. *15 Hz to 10 kHz Frequency Counter* . . . developed by implementing a very small amount of external circuitry (an FET, a transistor, etc.) and sampling an incoming audio frequency using the CPU's crystal-controlled clock.
4. *CPU Interrupt Handler Diagnostic*
5. *Memory Diagnostic* . . . right down to the bad chip.

THE GAMES . . . AND MORE GAMES – Computers are for kids (young *and* old), and DGSS has certainly put together an impressive collection of games written in Tiny BASIC. They have (at the time this was written, anyway) four cassettes of Tiny BASIC games . . . each cassette with five or more games. The number of programs prevents them from all being listed here, but some of the more popular ones are: CHOMP, CHECKERS, TIC-TAC-TOE, BIORHYTHM, TRAP, BLACK-JACK, 23 MATCHES, plus some good educational and simulation game programs. Also, available on a separate cassette are the game of KINGDOM and two versions of LIFE.

EDUCATOR-8080 – This is a very interesting program which demonstrates the effect executing certain instructions has on the status, Accumulator, and B and C registers of an 8080. All of them are displayed on the screen; you enter an instruction and then see the change. Quite a program for a teaching environment. (The only limitation was the fact that memory reference instructions couldn't be executed.)

The cassettes are reasonably priced at $5 to $10 . . . are recorded on high-quality tape . . . and are checked before shipment. Listings are not provided in an effort to keep the cost down . . . but

perhaps they can be purchased for additional bucks. All in all, it's a very impressive collection of software, and certainly one of the highlights of the system.

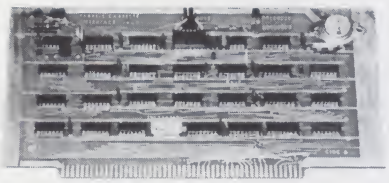
Summary

I like it! As I've pointed out, the various assemblies and peripherals which make up the system were easy to interconnect, and the available software gives you something to work and play with . . . now.

The Digital Group does not go along with the Kansas City Standard for cassette interfacing because it's too slow, compared to theirs. I don't consider this to be a major drawback, since it appears that an adequate amount of software will be available through DGSS. If you're going to be buying a system with the idea of doing a lot of software exchanging with others (who use the KC Standard), then you should keep this in mind.

By the time this article goes to press, the Digital Group should have among its offerings a selection of cabinets and keyboards, and most important, a very sophisticated dual Phi-deck cassette controller and drive system.

The system is not for everyone, because it isn't the least expensive on the market. I do believe, however, that you can be assured of getting your money's worth. (They don't have any qualms about selling *portions* of the system . . . as a matter of fact, they've published plans for converting an ordinary TV into a monitor . . . and, as I pointed out, you can probably get by with a less expensive cassette recorder.) If you would like to contact them for further information, the address is: The Digital Group, P.O. Box 6528, Denver CO 80206. They're very helpful folks, and I'm sure they would like to hear from you. ■



The Tarbell Cassette Interface

- Plugs directly into your IMSAI or ALTAIR
- Fastest transfer rate: 187 (standard) to 540 bytes/second
- Extremely Reliable — Phase encoded (self-clocking)
- 4 Extra Status Lines, 4 Extra Control Lines
- 25-page manual included
- Device Code Selectable by DIP-switch
- Capable of Generating Kansas City tapes also
- No modification required on audio cassette recorder
- Complete kit $120, Assembled $175, Manual $4

TARBELL ELECTRONICS

144 Miraleste Drive #106, Miraleste CA 90732
(213) 538-4251

California residents please add 6% sales tax

LIFE Subscription ?

Quite a few subscribers to Kilobaud have asked about a life subscription. For a limited time this will be available . . . as a convenience to readers who don't want to be bothered with yearly billing for subscriptions.

Inflation seems to be a way of life and is going to continue. The word is that paper costs are going to double in the next year, as will postal costs in the next year or so. This means that $15 subscriptions for magazines will go to $20, then $25 . . . and so forth. With a 5¢ (1940) ice cream cone now running over 50¢ and 20¢ magazines running about $2 per copy, in a few years we'll be used to paying $5 per copy . . . then $10.

The first copies of 73 were 37¢ and a lifetime subscription was $37. Now copies are $2 and the lifetime is $150 . . . and going up to $200 soon.

The plans are to sell no more than 1000 life subscriptions to Kilobaud. There are some good reasons for this, obviously . . . like those mentioned increases in costs. Don't be disappointed by waiting too long.

Time payments? Sure . . . send in $50 down and we'll bill you $25 per month for four months for the balance . . . we'll take cash, check, money order, BankAmericard, Master Charge, American Express. We're very easy to get along with.

What about your present three year for $25 subscription? If you act now (to coin a phrase) you can deduct that from the bill. Act now.

$ ____ enclosed ☐ cash ☐ check ☐ money order

☐ Paid in full ☐ Time payment - $50 down, $25 per month for four months

Charge ☐ BankAmericard ☐ Master Charge ☐ American Express

Card # _____

Interbank # _____ Expiration date _____

Signature _____

Toll free subscription numbers 800-258-5473 or 800-251-6771 1/77

kilobaud

peterborough nh 03458

I've heard several good comments regarding the SWTP printer since its introduction several months ago, and it looks like Denis Bourdeau is staying with that line in his article on the PR-40. His approach is quite objective and he does a good job of pointing out the good (mostly) and the bad. Should be of interest to anyone thinking of buying one and I'm sure the points Denis makes would apply to similar printers, also. — John.

How to Use the New PR-40 Printer

Denis R. Bourdeau
7700 West Glasgow Pl.
Bldg. 22, Unit B
Littleton CO 80123

Communicating with a computer is difficult enough without having to agonize over the purchase of peripherals. There isn't a single micro-mini-computer owner that wouldn't want a line-printer, disk-drive, and video terminal. Those who do not have eidetic memories of indefinite capacity are most likely to be very desirous of owning a hard-copy device. In particular, line-printers are very popular with software development types. In response to this demand, Southwest Technical Products Corporation has made available a practical line-printer for the hobbyist — the PR-40. Having purchased and assembled this printer, I would like to describe my experience with it.

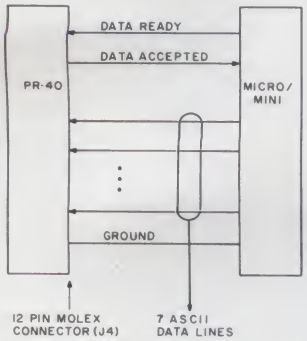


Fig. 1. Electrical Interfacing.

Features

Without a doubt, the most pertinent specification the printer possesses is its cost — $250. When one discovers that the PR-40 only prints 40 columns, the price somehow increases in impact. It wasn't until I realized how well it was designed that its purchase was easily justified. A dot-matrix, 75 line/minute printer possessing the 64 character ASCII subset ought to raise your eyebrows. Upon discovering that this printer also contained a 40 character line-buffer, I was impressed. The presence of a line-buffer means two important things:
1. The PR-40 can accept data fast (one character per micro-second), and
2. the driving computer can perform other functions while the line is being printed. (I'll get into hand-shaking in a bit.)

The PR-40 uses 3-7/8" adding machine paper which has the advantage of being common and cheap. Since a tractor-feed is not used, registration of special forms will be a problem. An ordinary typewriter type platen is used with perfectly adequate paper feed. Two other similarities to a typewriter are the platen/paper pressure relief lever for repositioning the paper and the use of a scarce but available 5/16" ribbon. (SWTPC says that they have extra ribbons. Also, 5/16" typewriter ribbon is available at office supply stores. I'm not crazy about the purple color of the SWTPC ribbons, but this is almost surely due to a prejudice originating from the experience of looking at grocery receipts. Note, however, that the ribbon can be turned over and that the paper width fits nicely on 8½ by 11" paper.)

When you see the PR-40 close-up and in action you will probably share my reaction: "Hmmm, looks cheaper than I expected. But look at how the printer-head works . . . and how the ribbon automatically reverses . . . and how the driving cam works . . . so few moving parts . . . reliable . . . hmmm, very clever! Yeah, should last a long time!"

The major reason that this

device is not considerably more expensive is the design of the cylindrical cam which drives the print-head. A teletype such as the ASR 33 responds to carriage-return and line-feed codes by returning the carriage and feeding lines. This is not the case with the PR-40. The cam is a cylinder grooved so that a tooth attached to the print-head drives the head from side to side as the cam rotates. The carriage-return code causes the line currently in the buffer to be printed when the cam rotation is underway. Thus no matter how many characters are to be printed, one rotation per line is required (that is 75 rotations per minute.)

```
----- PR-40 CHARACTER SET -----

 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]↑
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]↑
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]↑
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]↑
```

Fig. 2. Character Set.

Thus we see that Southwest Technical Products' PR-40 is an excellent compromise between speed and line length. Twice as many characters per line would slow it down intolerably (if you find teletypes doubly intolerable) and a true line-feed and carriage-return action would increase the cost to more than many would want to pay. Since the majority of my efforts are directed toward systems software development, 32 characters per line (on my video terminal) suits me about as well as 40 characters. One of my maxims is that there is always an application that illustrates an exception to any software-related rule. The author therefore has performed a significant amount of development with the exceptional PR-40.

In an effort to remain objective, it must be mentioned that one item concerns me: The check-out instructions caution one to not operate the printer for more than a minute or so if printing full lines. The reason given for this caution is a good one. In such a situation the 5 by 7 dot-matrix print-head solenoids tend to overheat. They are being pulsed with a hefty 40 Vdc. The instructions also specify the conditions which minimize this heating. A pot controls the solenoid timer so that if 400 microsecond "on" duration is achieved the best compromise between faint lines and overheating will be obtained. Since my scope does not do so well in measurements of time much less than milliseconds, I experimented with the adjustment until an easily read contrast occurred without unreasonable heating. This rough calibration was quick and simple due to the fact that one can easily access the intensity adjustment during the printing process. Even so, it is advised that those considering using this machine err on the light side of print intensity.

```
--- SAMPLE ASSEMBLER SOURCE ---
    (PRINTED ON THE PR-40)

;
;SUBROUTINE PRCHR TRANSMITS THE
;CHARACTER IN THE A-REGISTER TO
;THE PRINTER BUFFER
;
;THE CALLING ROUTINE WILL SUPPLY
;A CARRAIGE-RETURN (215) TO
;START PRINTING THE BUFFER
;
;(THE MSB OF THE A-REGISTER
;MUST BE SET FOR ALL CHARACTERS
;FOR PROPER PRINTER STROBING)
;
PRCHR OUT,007  ;ACCUM TO PORT 7
      XRI,200  ;TOGGLE MSB
      OUT,007  ;DATA-READY LOW
      XRI,200  ;TOGGLE BACK ON
      OUT,007  ;DATA-READY HIGH
;CHARACTER HAS JUST BEEN STORED
;IN PR-40 BUFFER
;
;WAIT FOR DATA-ACCEPTED LINE TO
;GO HIGH (MUST WAIT DURING THE
;ACTUAL PRINTING OF THE LINE)
;
WAIT  IN,002   ;DATA-ACCEPTED IS
      RRC      ;LSB OF PORT 2
      JNC WAIT ;LOOP BACK
      RET      ;DONE, RETURN
```

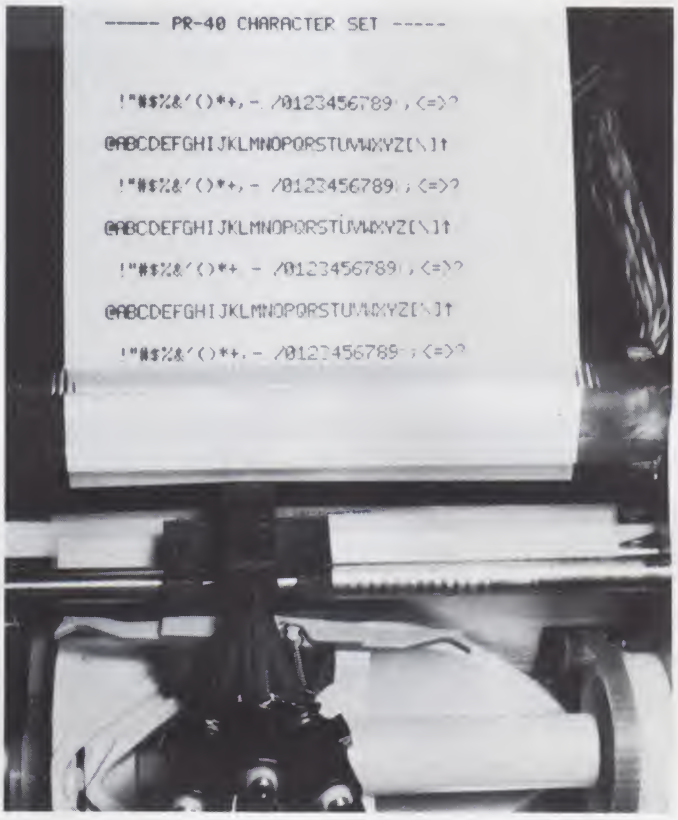
Fig. 3. Sample Program Printout.

There is one additional aspect to the mechanics of the PR-40 that deserves mention. At 75 lines/minute, each line of forty characters is zapped out at such a rate that a fair amount of noise is produced. The most dangerous effect of this noise is experienced when a cat is held near the printer just before it is started up. Yup, scares the h--l out of cats and I have long, straight scratches to prove it!

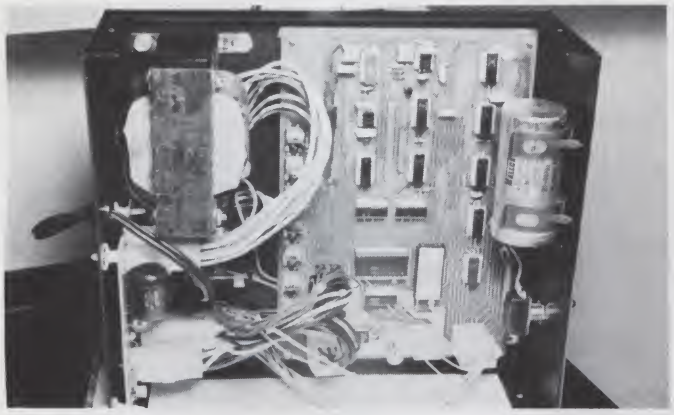


Author's highly customized Digital Group 8080 System.

A very nice feature of the ASCII character generator electronics is the conversion of small letters (octal 141 through 172) to capital letters. This really means that the electronics ignore the bit which, if set in the code for a capital, produces the small letter. Speaking of codes, a carriage return (CR) is a sacred character. Upon receipt of a CR the electronics initiates printing of the current line in the buffer and indicates to the interface that it is busy. A line-feed, as well as the other ASCII control characters are ignored — they don't even get into the buffer. With the Digital Group parallel interface card one can use seven lines for the ASCII codes and the MSB (8th) line for the "Data



Close-up of print-head, cylindrical cam, and character set.



Inside view of chassis.

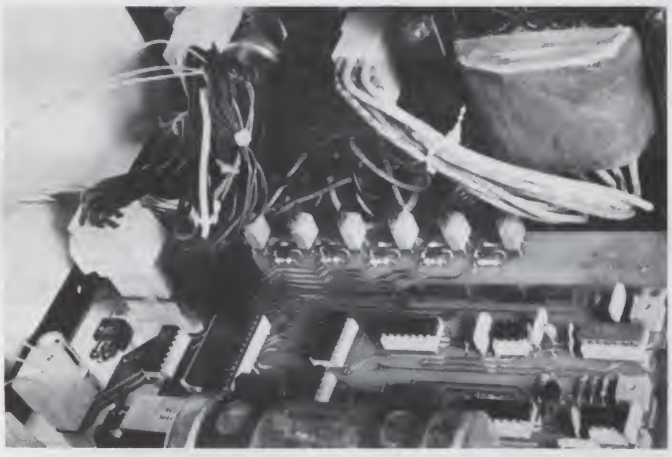
Ready" strobe to the printer. With the addition of two more wires, the "Data Accepted" line monitored by the driving software, and a ground, the interface is complete. Note that some bit of some input port (parallel) must be used to monitor the "Data Accepted" control line. The Digital Group System often has extra input port bits available.

Interface Software (i.e., the "drive")

The hand-shaking is simple and easy to code. The control lines are normally high and go low when active. SWTPC supplies a test program written for their 680 system. My 8080 system software developed out of a bit of experimentation and a lot of rereading of the single paragraph explaining the hand-shaking. Since few microprocessors can output data in parallel faster than once per microsecond, the transfer of data to the printer buffer occurs as rapidly as the software can be executed. You will only have to wait for the line to be printed — that is, if you have nothing else your software could be doing. If you do have other things going on, just be sure to check the "Data Accepted" line when reentering the printer driver.

The software involved in interfacing the printer is not very involved. At the lowest level, where the actual output instructions occur, the following sequence drives the parallel interface:

1. output the current character with the MSB set;
2. reset the MSB in the A register (use an exclusive-or-immediate);
3. output this result;
4. again toggle the MSB to complete strobing the printer logic (as in step 2);
5. output character;
6. enter a wait loop by first inputing from the port connected to the "Data Accepted" line;
7. check the corresponding bit (use the MSB or the LSB to allow a simple shift for testing carry bit);
8. loop back to step 6 until line goes high indicating
9. a return to calling routine.



Inside chassis — note interconnecting cables and print-head solenoid driver transistors.

The wait-loop is not exercised until the printer actually begins printing. You will notice that execution will fall through the wait-loop during transmission of the individual characters. This is due to the fact that the printer electronics have had more than enough time to stuff the character into the first-in-first-out buffer during steps 1 through 5. At higher levels, in the routines that call the described instructions, it is convenient to perform such functions as checking for character validity, translating line-feed (LF) and similar codes into CRs, and so on. In my Digital Group 8080 System, the operating system contains logic on a higher level yet which allows edited source information to be printed. In addition, it is very useful to dump memory, symbol tables, traces, simulations and so on to the printer. In fact, this article was drafted and edited with the use of the PR-40. This printer is serving a valuable function in connection with some elementary word processing. My word processing program simply columnarizes text to average 46 columns in width. It also makes a wild stab at hyphenation when it fails to find a blank to serve as the end of the line. In this way, articles can be typed directly from the PR-40 listings!

If you have a 32 character/line CRT terminal, you will find that 32 characters is quite adequate for software development purposes and that some commonality between CRT and printer I/O can be efficiently achieved!

Construction of the PR-40 is described fairly well in the supplied documentation. I must warn those who do not have a fair amount of previous experience with digital electronics assembly to be prepared to put in quite a few hours. You may even want to get some help. In my case, with a good deal of experience, it still took 20 hours to assemble the PR-40. One area of particular difficulty was the construction of the four main molex connectors. Not all of the connectors mated as snugly and as positively as they should have. This is a prime area for some innovation by using the individual connector pins rather than the nylon connector blocks. Some good heat-shrinkable tubing would allow the two J1 connectors to be simplified to pin-to-pin hook-ups. Only one connector had a real propensity for easily wiggling back out, so this is not a major concern. The connector blocks significantly reduce certain wiring errors, however.

An important piece of advice: This printer is complex enough to justify double-checking, triple-checking, and cross- and sideways checking!

The smoke-test is not a recommended method for checkout of this printer. The instructions specifically stipulate a testing procedure. This procedure is not complicated and will allow one to detect an error which could cause damage to the print-head. Essentially, measurement of the voltages fed to the print-head solenoids is required.

The second phase of checkout consists of adjusting the intensity and line-width trimmers. Should problems arise, you are on your own because SWTPC does not try to provide a comprehensive list of the most likely problems and the corresponding fixes. In fact, SWTPC says this, "In case of problems, the best procedure is to remove power and recheck all

assembly steps." One can obtain repair services from SWTPC, but any experienced kit-builder knows that the number of problems are inversely proportional to the care (double- and triple-checking) taken during assembly. My printer worked the first time, as did every other SWTPC kit I've ever ordered and built. In anticipation of maintenance, however, sockets were purchased for all of the IC's. Purchasing sockets for at least the larger, more expensive chips is highly recommended.

Lastly, with respect to checkout and use of the PR-40, the documentation contains an excellent explanation of the operation of both the printer mechanics and electronics. If the builder can program even a little bit, he can fairly quickly write the software interface and a routine to repeatedly generate the 64 ASCII characters for testing. The author spent a lot more time building than in interfacing — a ratio of about 5-to-1 to be precise.

Conclusion

The PR-40 was found to be unsuitable for special forms such as mailing labels. There is too little space between the print-head and platen for more than just the ribbon and a single layer of paper to occupy. Unfortunately, there aren't any adjustments possible which would permit the use of labels, multi-part forms, and the like. While it seems that a tractor-feed could be incorporated into the design, my feeling is that this would require a number of associated design changes which would be like putting landing-gear on a VW.

Those serious hobbyists, who may be constrained by money, space, and time — but certainly not ideas, should find that this microminiaturized hard-copy device will have a very significant effect on their productivity.■



LIGHT!

FAST!

DEPENDABLE!

WESTON 4449
Digital Multimeter

SPECIAL $149.

3½ digit. Solid state. Dual slope high impedance bi-polar A/D converter. Auto-blanking and polarity. Single chip for logic circuitry. Overload protection. **Ranges:** DC Volts: 0-199.9 mV — 1.999 V, .05% rdg ±1 digit. 0-19.99 — 199.9-1000 V, .1% rdg ±1 digit. AC Volts: 0-99.9 mV — 1.999-19.99-199.9 V, .3% rdg ±1 digit. 40 Hz — 10 kHz. 0-1000 V .5% rdg ±1 digit, 40 Hz — 2 kHz. 199.9 mV — 1.999-19.99-199.9 V, .6% rdg ±1 digit, 10 kHz — 20 kHz, 0-1000 V, 1% rdg ±1 digit, 2 kHz — 10 kHz. DC Current: 0-199.9 uA — 1.999 mA, .2% rdg ±1 digit. AC Current: 0-99.9 uA, .4% rdg ±1 digit, 40 Hz — 10 kHz. 0-1999 mA, .75% rdg ±1 digit, 10 kHz — 20 kHz. **Resistance:** 0-199.9Ω — 1.999 K, 1 mA @ 0.2 + 2 V, .25% rdg ±3 digits. 0-19.99 K — 199.9 K, 10 uA @ 0.2 + 2 V. .1% rdg ±1 digit. 0-1.999 M, 1 uA @ 2 V, .2% rdg ±1 digit. 0-19.99 M, .1 uA @ 2 V, .5% rdg ±1 digit. Specifications: **Conversion Rate:** 4 per sec. CMR: 80 dB. NWR: 38 dB. Power Req: 115 V, 50 — 400 Hz. Temp: 25°C 3°C, 0° C-50° C at derated acc. Size: 2.25"H x 5.45"W x 7"D. Weight: 2½ lbs.

INSTRUMENTS & COMPONENTS

AST/SERVO SYSTEMS, INC.
20 REPUBLIC ROAD, NORTH BILLERICA, MASS. 01862
617-667-8541

John Craig
Editor

Fire!

... let your micro call for help

This started out as an article dealing with the construction and use of single board kits (microprocessor-on-a-board) being offered by several manufacturers to the commercial and hobbyist community. Because of the "do-nothing" state these boards seem to be in after construction, I decided to put one to use in a demonstration of interrupts. Interrupt operation is going to become more and more important as we become more and more sophisticated with our personal computer systems; so if you're not familiar with their operation, let me suggest you come along for the ride!

The Single Board System — Is It for You?

There are a number of these boards available to the hobbyist today ... and the number keeps growing. The MPU (MicroProcessing Unit)-on-a-board has, in most cases, been developed for two primary reasons. First of all, they can *all* be used as part of a system ... either as a complete microcomputer system or as a dedicated controller (some with more effort than others). Secondly, several of these boards were designed and built with the objective of familiarizing the technician, engineer, or hobbyist with microcomputer technology, programming, and applications. The completed board provides the hobbyist (or engineer) with the opportunity to learn machine-language programming. Machine-language is, without a doubt, the least efficient ... but, in some applications (such as the dedicated controller) a small machine-language program may be all that is needed.

Regarding the first application ... there's no reason why you (the hobbyist) couldn't use one of these boards as the building block for creating a complete microcomputer. This could include a cabinet with front panel switches and indicators (and/or keyboard and display already on the board) along with a complete I/O system for interfacing with peripherals. (Although such an undertaking probably shouldn't be taken on by the newcomer!)

Regardless of whether the MPU-on-a-board is to be used as part of a complete system, or as a training device, the fact remains that you still have *only* an MPU when the board is finished. You should be prepared for a slight pang of disappointment when, after slaving over the construction of said board, you apply power for the first time. (I didn't have the radio on ... but in the background I could hear Peggy Lee singing, "Is that all there is?") It's doubtful there are many things as *dull* as a computer what don't got no peripherals! They have a habit of just sitting there and not doing much. But then, they aren't expected to do much ... after all, it is just the MPU.

One board I built recently had a cute (?) little counting program to demonstrate the operation of the board. The program outputs to the double digit display a count sequence (in decimal, even) from zero to 99. You can make it count faster ... or make it count slower ... or, if you're in the mood for a lot of variety, you can make it count slow *and* fast.

Let's face it ... the real fun with a computer begins when you put in a worthwhile program and make it start doing something. Actually, the mechanics of entering a program is kind of a kick ... whether it be from a cassette,paper tape, TTY, TVT, or any other input device. And of course, getting a print-out on some kind of output device is something we all desire, too. With most of these boards, the input device is a hexadecimal or octal keyboard with several command keys. The output device usually consists of two or more indicators for displaying the hex or octal digits.

Now ... perhaps this has all sounded somewhat negative up to this point. (In other words ... you've only got this, but it would sure be neat to have all that!) Not so. One of the real challenges for the hardware types in this hobby is the interfacing of different devices to the computer. And, you've certainly got the opportunity to meet that challenge with one of these boards. So ... why not start with a MPU, a monitor program, some RAM, an input and an output port ... and take it from there?

The board I mentioned a moment ago uses the MOS Technology 6502 processor chip. I'd like to keep the following discussion in a general direction, rather than get into specifics, so we're going to use this 6502-based board as a *typical* MPU-on-a-board system (and the word

"system" is being used loosely here).

The construction of the board was a breeze. As a matter of fact, the only problem encountered was when it came time to plug the 6502 MPU chip into its socket. It's a 40-pin package . . . and all 40 pins were a bear to get in! But, once that was accomplished and the initial power was applied (+5 V and -9 V), it sprang to life with vim and vigor! (Actually, it just sat there . . . but, at least it didn't smoke.) (*Note:* After installing the IC sockets and cleaning the board, I made a good visual inspection to ensure there weren't any solder bridges. Then I applied power to the empty board (i.e., no ICs) as a further check against shorts.)

The Hardware and Software

The MOS Technology hardware and software manuals supplied with the board furnished me with the information I wanted when I went to it . . . but, I can't really give an evaluation of the programming manual because it was written for assembly-language programming (with virtually no machine-language examples . . . which is understandable). The manual was written for a programmer who is going to be developing programs for the 6502 using the Resident Assembler or a Cross-Assembler (i.e., an assembler for the 6502 mnemonics (symbolic code) which was written to be run on a different, and usually larger, computer).

This particular board came with a monitor program (256 bytes) in PROM with three additional sockets for adding more PROMS. The monitor program is for loading, executing, and displaying programs in the 1K of RAM memory. A function I didn't care for in this monitor (and, which caused me to have a software bug) was the fact that when you enter an address into memory (16 bits — 4 hex digits) the *least significant* portion is entered first. For example, if you wanted to put in a LOAD A from location FA05, the op code for the LOAD A (AD) would be entered . . . followed by entering "05" and then "FA". Tis a bit confusing.

After loading in the demonstration program, I called in the wife and kids and said, "Look, there's a whole computer right there on that one board! It doesn't do much, except sit there and count . . . ah, but wait a minute . . . I can change this location (What's a location, Daddy?) and make the thing count slower or faster. Pretty neat, huh? Just think . . . a whole computer on that one little board!"

Well, I don't have to tell you just how *impressed* those folks were! (Actually, as you may have guessed, I'm still trying to convince myself just how impressed they were!) So . . . it was "back-to-the-old-drawing-board" time. If not for their benefit, I decided that at least for my own, I was going to make this thing do *something* besides sit there and count.

Interrupts?

In a rare flash of brilliance I came up with a very simple circuit for demonstrating an *interrupt* operation. I happened to have (laying around in my junk box) a heat sensor . . . which I'd never found a use for. With this I built a circuit to generate an interrupt to the processor when this sensor was activated (by holding a match under it . . . i.e., a "flame detector").

An Interrupt Application

I've heard the remark, "So what?," with regard to interrupts and the home computer. There are those that feel interrupt operation isn't going to be a big consideration with the home system, but I can assure you that as things get more and more sophisticated (which they will) interrupts will be used more and more. It's really a very efficient way of doing things.

Let me illustrate this "efficiency." We can either have our computer sitting around *waiting* for a particular input (i.e., an external event), or we can have it busy processing data and doing a number of other things and *interrupt* it when that external event occurs. Perhaps the most common example we could use would be our friend, the noisy Teletype machine (or, if you're lucky, you've got a friend who is a TVT). If we're running a program which depends on inputs from the TTY (*or* TVT) we can either have the processor sitting in a loop waiting for that input . . . or we can have it doing other things and then interrupt it when a key is hit. (And, eventually the home computer *will* be doing other things besides running your one program.) The interrupt will force the processor to exit the program being executed at the end of the current instruction. From there it will go and execute a subroutine for handling the interrupt (i.e., a check will be made to see which key was hit on the TTY . . . and what action, if any, is to be taken). After executing this subroutine a return will be made to the program which was interrupted, and execution of that program will continue (as though the interrupt had never occurred).

Here's something else to consider with regard to interrupt operation. How many TV sets do you have in your home? If you're a typical American family you've got at least two. In the years to come we're going to find that one terminal for the computer isn't going to satisfy the demands made by your family. You're going to be wanting to work with it developing software, or whatever, and someone else in the family will be wanting to run an educational game or accounting program. The answer, of course, will be to have two terminals . . . running in a time-share mode . . . which will be based on interrupt operation. Think about it. (And, excuse the use of the word "work" up there . . . very bad.)

Now, if I've convinced you of the need to become familiar with, and use, interrupts (assuming the "convincing" was needed), let's get on with a discussion of an interrupt circuit and how it was implemented.

Fig. 1 is a schematic of the circuit which I built to demonstrate an interrupt using the board. The MOS TECHNOLOGY 6502 has two different interrupts. One is maskable through software (IRQ) and the other is a nonmaskable interrupt (NMI). The operational requirements for the IRQ were somewhat more stringent than the NMI . . . so I went with the easier NMI. The nonmaskable interrupt is edge-sensitive and is simply

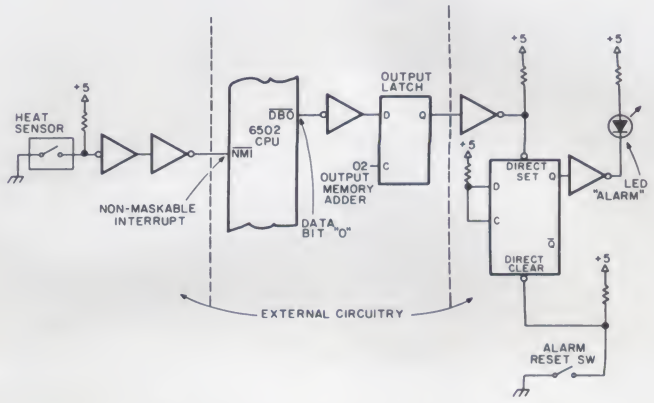


Fig. 1. Interrupt circuit schematic.

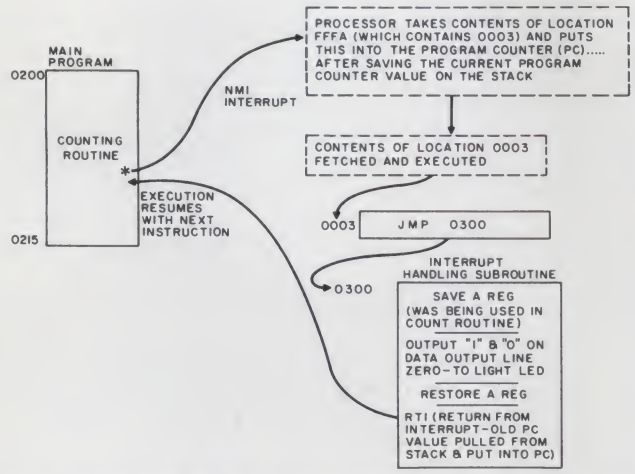


Fig. 2. Block diagram of main program and interrupt handling subroutine.

LOC	CONTENTS		COMMENT
0003	4C	JMP TO 0300	JMP
	00		(TO INT SUB)
	03		
0300	48	PHA	
1	A9	LDA (IMM) 01	SET OUTPUT LATCH, BIT "0" (AND SET LED F/F)
2	01		
3	8D	STA (ABS) F900	
4	00		
5	F9		
6			
7	A9	LDA (IMM) 00	RESET OUTPUT CATCH, BIT "0"
8	00		
9	8D	STA (ABS) F900	
A	00		
B	F9		
C	68	PLA	
D	40	RTI	RETURN FROM INTERRUPT

Fig. 2(a). Interrupt handling subroutine.

looking for a one-to-zero transition.

When a match is held under the heat sensor it eventually reaches a certain temperature and the contacts close. This causes the NMI line to drop low. The program which was being executed is interrupted at the end of the current instruction and the processor then begins executing an "interrupt handling" subroutine. In this subroutine, Data Bit 0 is set to a 1 and sent out to the Data Bus Output Latch. (This is accomplished simply by storing a word — with bit 0 set — in memory location F900, which is dedicated to DATA OUTPUT OPERATIONS.) As a result of storing the word in this particular location a clock pulse for the Output Latch is generated, and the 1 is clocked into bit 0 of the latch. The Output Latch being in a set condition generates a Direct Set to the external alarm flip-flop. This, in turn, activates an LED which is the alarm. (The LED is used to provide an indication that the interrupt handling subroutine had been executed. Naturally, it wouldn't make a very good fire alarm!) Sending a 1 to the Output Latch is followed (in the subroutine) by outputting a 0 . . . so the latch will not remain in a set condition. This allows me to reset the alarm flip-flop using the switch coming in on the Direct Clear input. (I haven't provided pin numbers and parts identification on this schematic because it is of such a specialized nature . . . and I doubt that anyone will be building one.)

Fig. 2 is a block diagram of the software involved in this operation. As I mentioned earlier, the board came with a demonstration program which counted from zero to ninety-nine. I used this program as the *executive*, or main program, which was to be interrupted.

Upon recognizing the interrupt, the Program Counter is pushed onto the stack, and then the processor takes the contents of location FFFA (which contains 0003) and places that value in the Program Counter. (Location FFFA is referred to by the programming manual as a *vector pointer* because it is used to *point* to the actual interrupt subroutine address.) The Program Counter (PC) now contains 0003, and the JUMP instruction in that location is fetched and executed. The JUMP is to location 300, which is the beginning of the Interrupt Handling Subroutine. (The jump to 300 is part of the programming which you, the programmer, must do. The previous steps are done automatically by the processor . . . and the "0003" value is a result of the monitor program.)

The first item of business in the Interrupt Handling Subroutine is saving the A Register by pushing it onto the stack (because it was being used in the counting program). Then a 1 and a 0 are stored into location F900 (DATA OUTPUT location). The alarm flip-flop is set, and the alarm LED is lit. Then, the A register value is pulled off of the stack and restored into the A register. The last instruction in this routine is a Return from Interrupt (RTI) and its function is to restore the Program Counter (using the value pushed onto the stack in the very beginning, that is, the address of the next instruction to be executed in the program which was interrupted). The instruction at that location is fetched and executed . . . and the main program resumes as though nothing happened.

Naturally, after building this fantastic circuit I just had to demonstrate it for someone . . . anyone! You can see



Demonstrating the flame detection circuit

SUNTRONIX COMPANY

CHRISTMAS PARTS SALE

All parts are BRAND NEW PRIME UNITS unless otherwise specified. Some are limited quantities. These prices are effective ONLY thru December 31, 1976.

BRAND NEW PRODUCT FROM SUNTRONIX. Did you ever wonder how the pros were able to etch a complicated PC board with such outstanding results? Do you want to be able to etch a HI-DENSITY PC board and still keep all those FINE traces intact? The secret is in the type of etching equipment used. BUBBLE ETCHERS are many times superior to hand agitated trays or tanks — better than most spray etchers and inferior to none. We are now manufacturing a line of BUBBLE ETCHERS that are of first class quality, designed with the hobby and semi-professional application in mind, and VERY INEXPENSIVE. We offer several models of basic ETCHERS as well as a wide range of accessories, such as PC card holders, air pumps, plumbing, immersion heaters, etc. Prices start at $69.95 for our unheated model, which includes one PC card holder. Please write for more information.

● PC BOARD STOCK. First grade epoxy glass 1/16" cut to your size for only $0.02 per square inch, single sided. Double sided $0.035 per square inch.

● IMMERSION TIN PLATE SOLUTION. For professional PC boards you must tin plate them. Enhances solderability and appearance. 1 qt. will plate dozens of average size PC boards. No fuss, no muss — dunk 'em and out come bright shiny easy to solder PC boards. Instr. incl. $8.95/qt.

● VIATRON SYSTEM 21 (BRAND NEW UNITS). Checked out and operable, but sold as-is. Only 5 left so act fast. Shipped freight collect for ONLY $295.00. (Check previous ads of our competitors and be AMAZED.)

● COPPER ETCH CRYSTALS. Dry powder mixed with water forms a very fast and safe copper etch solution. Easy to dispose of when exhausted. Enough to make three gallons of etch. 5 lbs. $4.95 ea.

● PC BOARD PROJECT KIT. This kit includes an assortment of single and double sided PC stock in useable sizes, plus one pint of immersion tin plate solution and five pounds of etch. Instr. included. $14.95 ea.

● MULTI VOLTAGE REGULATOR CARD. Removed from functioning power supplies and guaranteed. Complete regulator for +15, -15 and +5 volts. Currents in excess of 3.0 Amps. You supply the raw dc voltages and a case; we supply the rest, including the pass transistors good for 10.0 Amps. Schematic included. $14.95.

● COPPER CLEANER SOLUTION. Immersion removes soil and oil. $5.95/qt.

● FLAT RIBBON WIRE. 20 conductor #28 Ribbon at give-away prices. $0.39/ft. Min. order, 10 ft.

● TRANSISTORS

2N3859 Equiv.	NPN	$.20
TIS93	PNP	.49
TIS98	NPN	.59
MPS2222A	NPN	.20
MPS2907A	PNP	.20

● LINEAR

709 Op Amp	TO-5	$.20
301AN	DIP	.49
DATEL IC8B	DAC	9.95
LM309K	TO-3	1.49
NE555	DIP	.50
LM5000	Reg	4.95

● UART. COM2502/2017 40 pin DIP UNTESTED. $3.95. But the ones we've checked work fine at 4.5 volts instead of 5.0.

We also stock an in-depth line of 7400 series TTL, including CMOS. Please inquire as to availability and price. Video Display Terminal subassemblies and Keyboards are still available, but the supply is dwindling. Graphics Drivers are in full production and available from stock. Please see our ads in August and September 73 for details.

TERMS: Full cash price, plus shipping costs MUST be included with order. We accept MasterCharge and BankAmericard. Please, NO CODs. Excess shipping payments refunded promptly. Prices and availabilities subject to change without notice. NO EXPORTS.

SUNTRONIX COMPANY
360 Merrimack Street, Lawrence MA 01843 617-688-0751

this actual, true-to-life demonstration being performed for my daughter Sheri in the photo. Light the match . . . hold it under the sensor . . . watch the LED come on . . . and the processor never misses a beat in its counting routine. Thrilling! (I jabbed her in the ribs a moment after that picture was taken . . . and found that she was sound asleep!)

A Practical Fire-Detection System

The circuit described in the previous section is just fine and dandy as an example of how an interrupt works using a heat-detection circuit. But, it certainly has some serious drawbacks as a practical fire-detection circuit. First of all, the computer is instrumental in setting off the alarm in the *flame detection* circuit. I think that would be a mistake. As much as I love and respect my computer, I wouldn't want it to be responsible for saving my life! I have two very good smoke detectors installed in my home which set off a very loud alarm when smoke is detected. (This is, of course, the other obvious shortcoming of my *heat*-detecting circuit . . . *smoke* detection is the only way to go.) The smoke detectors work off of batteries which is another advantage, because there is always the chance that the fire might be started from an electrical overload or short, which might be on the same circuit as the computer . . . ! Also, with regard to the computer, it would be just my luck that it would get hung up in a loop (the wrong one) on the night my house decided to burn down. (Although it might not be a bad idea to have the computer monitor the batteries in the smoke detectors and sound an alarm if they get too low.)

I think the most practical part the computer could play in a fire-detection system would be to call the fire department and alert them to the fact that you have a fire in your home. Fig. 3 is a block diagram of my proposed system.

Before we get into a discussion of the hardware and how it could be implemented, let's take a look at the *situation*. In other words, let's see if we can examine your thoughts and actions if you were suddenly awakened in the middle of the night by an alarm from a smoke detector. Your first two immediate concerns would be getting your family and yourself out of the house and finding out where the fire is. It's rather doubtful that you would want (or, in many cases, be able) to stop and phone the fire department. I personally would want to get to that garden hose and start doing what I could to stop the fire.

You could run to a neighbor's house (or send your wife, if ya got one) to call the fire department. *But*, the beauty (?) of my proposed system is that the fire department would be notified and probably on their way by the time you get out of your bedroom! Those first few minutes can certainly be important ones, too. The big factor to consider is this: How long can you hold your breath? It's smoke inhalation that kills people. If a member of your family has been overcome by smoke, then the minutes it takes the fire department and their oxygen to arrive may mean the difference between life and death.

Another feature which I consider important would be to have a separate audio alarm (perhaps a bell . . . something different from the smoke detector's) which would tell you that the computer had successfully contacted the fire department and that they acknowledged and were on the way. If you heard this alarm you could go ahead and do what you could to fight the fire, comfortable in the knowledge they were coming. On the other hand, if you didn't hear it you would know that a call was still needed.

The hardware for implementing such a system as this is available. We're not talking about a wild dream . . . just something that would very likely be quite a hassle to build and implement! In the block diagram you'll notice that I left the smoke detector intact and simply brought out the signal which sounds the alarm. This signal will be used to generate an interrupt to the computer (which will be running in the executive, or main program, of your Home Operating System).

This interrupt signal will be enabled to the processor by either a manual switch or a timer. The switch will be thrown to the enable position if you're going to be away from the house for an extended period (day or night). For example, you could have the timer enable the interrupt signal only from 9 pm to 7 am. This is done because the smoke detectors can sometimes be set off inadvertently by someone smoking underneath them. Therefore, it might be better to have the system off during the day to eliminate the possibility of false alarms (which the fire department certainly wouldn't appreciate . . . *especially* from a computer). Of course, this problem (of disabling during the day) could be eliminated by banning smoking in the house, right? On the other hand, maybe there's nobody around your house during the day.

The timer circuit could actually be a software routine or, even better, a real-time clock in the computer.

This is, of course, a trade-off decision which must be faced whenever either hardware or software is being designed. There are a lot of considerations. For example, is the computer going to be sitting in its timer software routine all the time, or will it be used for other chores as well as the timekeeping?

Note that the interrupt signal is going into an Interrupt Priority Network. This circuit will allow you to have a number of interrupts from different sources (which will very likely be the configuration for the eventual home system). In this case the computer will either poll the interrupts to find out which

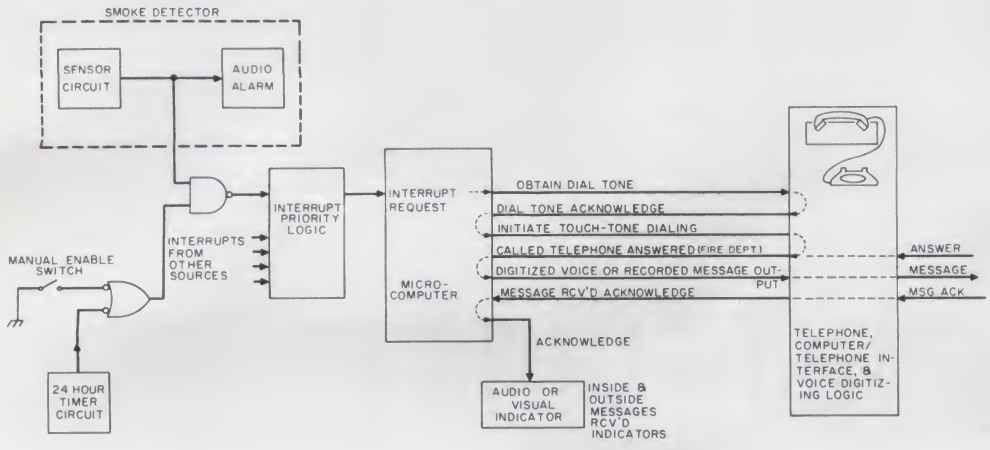


Fig. 3. Block diagram of a practical fire detection system.

one is active or will use interrupts of the vector type. But, that's a subject for another article. (Anyone care to bite?)

The heart of this system will be a program (and hardware) which will generate a digitized voice message once the fire department has answered the phone. (It would also be quite possible to use a regular portable cassette recorder . . . but, I would think it would then have to be dedicated to this application.) The sequence of events performed by the computer is depicted by the lines coming out, from top to bottom. (The sequence of events represents both hardware and software operations.) After obtaining a dial tone the interface logic sends an acknowledge to the processor, which responds by initiating the touchtone dialing of the fire department's number. When the ringing stops (i.e., when the phone has been answered) the computer begins execution of the digitized voice message program . . . "This is a computer-generated message to inform you that a fire has broken out in the Smith residence which is located at 501 Anchor Way. To repeat, this is a computer-generated message to inform you that a fire has broken out in the Smith residence which is located at 501 Anchor Way. If you acknowledge receipt of this message, please flash your receiver three times." Logic which would detect these three flashes would generate an acknowledge signal back to the computer. The computer would in turn activate the "message received" alarm/indicators both inside and outside of the house.

Without a doubt, were one to build a system such as this, it would be advisable to tell the local fire department about it . . . and perhaps even demonstrate it for them. Also, it's very likely that there are some considerations which I've overlooked. If you have any comments, I'd be interested in hearing them. ■

All an **ASR 33** is and **more!**



OLIVETTI 318 TELETYPEWRITER
BUILT-IN PAPER TAPE I/O
HEAVY DUTY DESIGN
ELECTRIC TYPEWRITER-STYLE KEYBOARD
EXTRA 10 KEY NUMERIC PAD
FRICTION OR SPROCKET FEED
SUPPORTED BY OLIVETTI

$950 + SHIPPING

GREEN PHOSPHOR MONITOR $150

$10 SHIPPING

1V P TO P COMPOSITE VIDEO INPUT
LONG PERSISTENCE FOR GRAPHICS
10 MHz BAND WIDTH
ANTIGLARE 7x9" SCREEN
RASTER SCAN
NEAT TABLE TOP UNIT



SwTPC 6800...$395
IMSAI 8080...$599
ONE DAY SHIPMENT

COMPUTER WAREHOUSE STORE



WRITE FOR OUR CATALOG
Dept. B
584 Commonwealth
Boston, MA 02215
"THE BIGGEST COMPUTER HOBBY CENTER IN THE WORLD"

SOFTWARE

6502

IN RESPONSE TO POPULAR DEMAND TSC HAS WRITTEN SEVERAL PROGRAMS FOR THE USERS OF 6502 BASED COMPUTER SYSTEMS. THIS PACKAGE CONTAINS FIVE OF OUR MOST POPULAR GAME PROGRAMS AND IS COMPATIBLE WITH KIM, TIM, OSI, AND JOLT MONITOR SYSTEMS WITH AN I/O TERMINAL. YOU GET EXCITING VERSIONS OF HANGMAN, ACEY-DUCEY, SWITCH, MASTERMIND, HURKLE, AND EVEN A RANDOM NUMBER GENERATOR, ALL BOUND IN A HANDY BINDER. THIS ASSEMBLY LANGUAGE SOFTWARE PACKAGE INCLUDES COMPLETE USER DOCUMENTATION. YOU GET A COMPLETE, WELL COMMENTED, ASSEMBLED SOURCE LISTING, INCLUDING A SORTED SYMBOL TABLE AND HEX CODE DUMP, INSTRUCTIONS FOR USE AND EVEN SAMPLE OUTPUT. HOWEVER, NO PAPER TAPES OR CASSETTES ARE AVAILABLE AT THE PRESENT TIME. THIS PACKAGE IS EXACTLY WHAT YOU HAVE BEEN WAITING FOR. AND ITS ONLY $19.95. ORDER PD4

8080

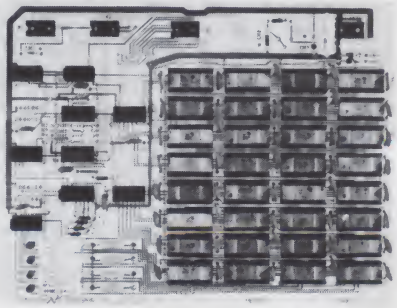
ATTENTION 8080 USERS. A PACKAGE SIMILAR TO THE ONE DESCRIBED ABOVE WILL SOON BE RELEASED FOR 8080 MICROPROCESSORS. WATCH FOR OUR ADS.

6800

NOTE THAT TSC ALSO HAS OVER 20 PROGRAMS FOR 6800 SYSTEMS NOW AVAILABLE. SEND $.25 FOR A COMPLETE SOFTWARE CATALOG. WHEN ORDERING, PLEASE INCLUDE 3% FOR POSTAGE. INDIANA RESIDENTS ADD 4% SALES TAX. CHECKS WILL CLEAR.

TSC TECHNICAL SYSTEMS CONSULTANTS TSC
BOX 2574 W. LAFAYETTE INDIANA 47906

• 270 nsec Access Time • 470 nsec Read/Write Time • TTL Compatible Address Bus • Tri-State Data Bus Driver • Fully Socketed • Sphere Compatible • Easy Home Brew Interface • Voltages +12, +5, -5 •



LOW COST MEMORY 16K x 8 BIT DYNAMIC RAM

Model	Description	Price
WWW-16KA	Fully Assembled	$650.00
WWW-16KK	Kit	$550.00

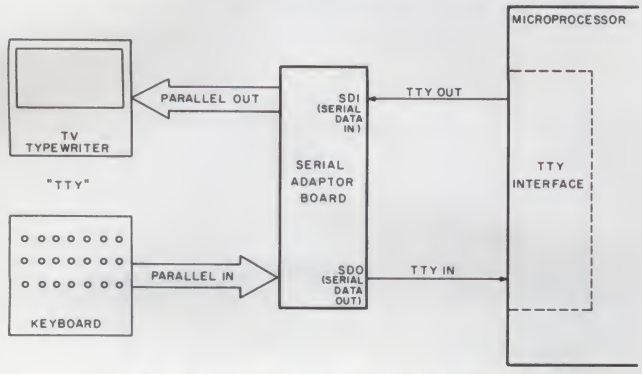
WWW ENTERPRISES
P.O. Box 548,
Harbor City CA 90710
(213) 835-9417

Robert Grater
1595-21 Laurelwood Rd
Santa Clara CA 95050

A Teletype Alternative

Almost every single-board computer we see these days was originally designed for the industrial community and snatched up by the hobbyists when and if the price got right. Many of these boards have TTY interfaces, which, in most cases, are not too useful to the less affluent hobbyists. This is a shame, because that input and output are being wasted along with the routines in ROM for driving them. But, all is not lost! Bob Grater has the answer for using that TTY I/O . . . the Serial Adaptor Board. This article should appeal to those of us with TV Typewriters (vs. video boards from Proc Tech, Digital Group, Polymorphic, etc.). If you build the board (from scratch) or purchase and interface it with a processor not mentioned in the article, I'm sure the manufacturer would appreciate a copy of your interface scheme to share with others. — John.

A number of the new microcomputers now on the market are designed to fit the novice computer buff's pocketbook, teach him machine language, and act as the basic CPU for an expandable system. Many of these microcomputers have serial TTY ports already on the main board. Somehow this always seemed a bit inconsistent to me, since an ASR-33 will run the better part of a kilobuck, requires a good bit of maintenance, and does strange things to domestic tranquility when operated at 4 am on a Sunday morning! Most of these microcomputer-on-a-board units were designed for commercial applications and the companies doing the development work can certainly afford ASR-33 TTYs. Not so with the hobbyist!



Serial Adaptor Board, Block Diagram

The Serial Adaptor Board (SAB) described in this article won't help with the problem of acquiring an ASR-33, but it will allow accessing the TTY port on a microcomputer, therefore taking advantage of the routines already in ROM for TTY use. Too slow, you say! Not really. KIM, for example, will run at 300 baud (and faster, but I'll stick to manufacturer's specifications) and SWTP 6800 will go at 1200 baud.

The SAB-1 Serial Adaptor Board was designed to be as universal as possible when interfacing with a parallel TV typewriter to a computer serial TTY port. It features an adjustable on-board clock, Serial Data Out (SDO), not (indicated by overbar) Serial Data Out ($\overline{SDO}$), Serial Data In (SDI), $\overline{SDI}$, KS and $\overline{KS}$ Keystrobe Out to the TVT, and jumper selectable programming of the I/O plugs.

The SAB-1 is built on a 3" x 3½" PC board using a NE555 as a clock oscillator. Baud rates between 90 and 460 are available within the range of the 20k trimpot. Higher baud rates can be obtained by simply exchanging the 0.01 uF precision polystyrene capacitor for one of a smaller value. Also, an external clock may be used by lifting the jumper from pin #3 of the NE555 and running the external clock into the clock line.

Parallel to serial data conversion is done via the AY-5-1013 (or similar) UART which is run in an asynchronous unconditional mode. The different data inversions available are accomplished by the CD-4049 CMOS Hex Inverter/T^2L Converter (the outputs of which will drive two standard T^2L loads).

Power Requirements

The +5 volt supply draws 20 mA maximum in all cases, and the current requirement depends on the loads driven by the 4049. The +5 volts may enter the board at either J1 (TVT) or J2 (KYBD). The -12 volts is required only for the AY-5-1013 UART. If a AY-5-1014, 6402, or some other UART is used, the -12 volts will probably not be required. The -12 volts enters the board at J1 (TVT). Extensive filtering is done on the board to ensure transient suppression.

Keyboard Connections

The keyboard interface (J2) requires data bits 1 through 7 to be positive going and the keystrobe to be negative going (KS). Pulse widths are not critical since the UART will accept them as they come along. If your particular keyboard has a positive going keystrobe and you are not using *connections* A and B on the board to invert SDI, you may invert KS in this inverter section. A positive KS in at point B will give you $\overline{KS}$ at point A, which may be tied to pin 2 of J2 for $\overline{KS}$ in. Bits 1 through 7 are jumper selectable to pins 9 through 16 of J2 (see construction section) depending on what configuration you wish.

TVT Interface

Data Out bits 1 through 7 are jumpered to J1 pins 9 through 16, again depending on the configuration you desire. The keystrobe is available at pin 7 of J1, and may be either positive or negative going (KS or $\overline{KS}$) depending on the jumper selection at KS

SEL. All power supply voltages are normally fed to the board via J1 since most TVTs have the -12 volts available. The +5 volts is run across the board with heavy bypassing so your keyboard may be powered directly from connector J2.

Board Construction

1. Normal PC board build-up procedures are used following the parts placement diagram*. Permanent jumpers (8 total) are installed at the points indicated by the arrows on the layout diagram. The jumper on J2 pins 3, 4 and 5 to ground is first run through one of these pin connection holes for the socket, then through the hole in the edge foil in the board (ground).

2. *Keystrobe Out SELECT:* The KS output to the TVT is selectable; for a positive going KS, the point marked SEL is jumpered to the point marked KS. For a negative KS, SEL is jumpered to the point above KS.

3. *Serial Data Out:* SDO (U2 pin 2) and $\overline{\text{SDO}}$ (U2 pin 4) are always available; either or both may be used for your system.

4. *Serial Data In:* SDI is connected directly to the point marked SDI. If SDI is required for your installation, point SDI is jumpered to point A, and the $\overline{\text{SDI}}$ signal enters at point B.**

5. *Data Bits:* Data Bits are jumper selectable at both J1 (TVT) and J2 (KYBD). Please note that UART (U1) pins 5 and 33 are brought out to jumper select pads; ignore these as they are not connected internally in the UART. So only 7 of the 8 pins available on the UART side of both J1 and J2 are used. Data bit outputs from the UART are shown as follows:

TX Data to J1

Bit	UART Pin #
1	12
2	11
3	10
4	9
5	8
6	7
7	66

RX Data from J2

Bit	UART Pin #
1	26
2	27
3	28
4	29
5	30
6	31
7	32

6. *Power:* The board draws little power (20 mA @ 5 volts and 5 mA @ -12 volts); this is brought up via J1 from the TVT. The 5 volt line runs through to J2 with heavy bypassing so your keyboard may be powered via this single plug.

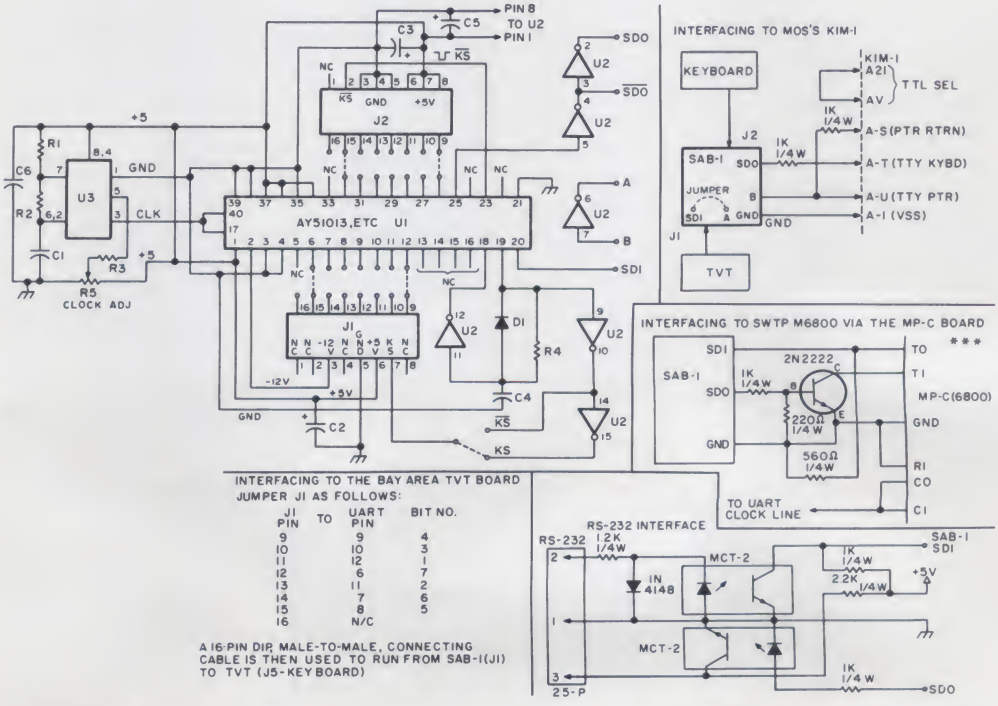
7. *Clock Adjustment:* The clock is available at pin 3 of the NE555 (U3). A counter is handy for setting it up but is not mandatory. With the components furnished, clock frequency is adjustable for baud rates between 90 and 460. The baud rate may be determined by dividing the clock frequency by 16. Thus, 110 baud (TTY) would be 1760 Hz. For KIM, this is no problem since it is self-adjusting. If you don't have access

Symbol	Quantity	Description
U-1	1	General Instruments AY-5-1013 (or similar) UART
U-2	1	CD4049 Inverter Driver
U-3	1	NE555 Timer
R-1	1	4.7k, 1/4W Resistor
R-2	1	56k, 1/4W Resistor
R-3	1	1k, 1/4W Resistor
R-4	1	100k, 1/4W Resistor
R-5	1	20k Trimpot
D-1	1	1N914 Signal Diode
C-1	1	0.01 uf Polystyrene
C-2,-3	2	10 uf @ 25V Tantalum
C-4,-5,-6	3	0.01 uf ceramic disc
Misc. Sockets	3	16 Pin DIP sockets (J1, J2, & CD 1049)
	1	8 Pin DIP socket (for NE555)
	1	40 Pin DIP socket (for UART)

SAB-1 Parts List

J1 (TV Terminal) Pin	Conn.	J2 (Keyboard) Pin	Conn.
1	N/C	1	N/C
2	N/C	2	Key Strobe
3	-12 volts	3	- (Gnd.)
4	N/C	4	- (Gnd.)
5	- (Gnd.)	5	-(Gnd.)
6	+5 volts	6	+5 volts
7	KS or KS	7	+5 volts
8	N/C	8	+5 volts

Pinout listing for nonselectable pins on J1 and J2

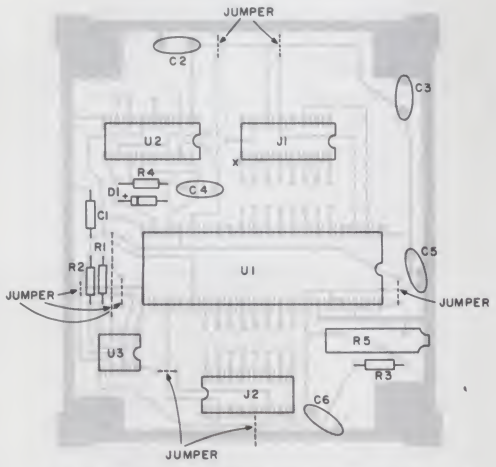


***Do not install the jumper between the 555 (pin 3) and the UART clock line (UART pin 40-17); instead, connect the clock line to the jumper at C1-C0 of the MPC Board.

Serial Adaptor Board, Schematic Diagram

*For Bay Area TVT, the socket at J1 is installed so that pins 9-16 face the jumper pads from the UART (U1).

**If this inverter is not used to invert serial data in, it may be used to invert a positive incoming keystrobe by bringing the KS directly to point "B" and jumpering point "A" to pin 2 of J2.



Serial Adaptor Board, Component Layout

to a counter, clocking into other systems requires a "turn and try" procedure. Adjust the pot until you get solid write-through, then go both ways until the data drops out, finally arriving at a center setting between these points.*

Interfacing with KIM

Interfacing with the TTY port on the KIM-1 is quite easy. It allows you to use the teletype routines already in ROM, including the TAPE DUMP which allows display of any series of memory locations. Two 1k, ¼ W resistors and a SPST switch are required if you wish to select between your TVT and the KIM onboard display.

On the SAB-1, point SDI is jumpered to point A, then point B is run to pin A-U on the KIM applications connector. A 1k pull-up resistor is installed between pins A-U and A-S. Point SDO is run through a 1k, ¼ W resistor to pin A-T, and ground is brought from the edge foil on the SAB-1 to pin A-1. Pins A-21 and A-V may be permanently jumpered to allow TVT operation, or a SPST switch installed to allow access to the on-board display.

With everything connected, press the RS (reset) button on the on-board keyboard. This gets us back into the monitor program. Then type a RUB OUT (ASCII 1111 1111) on your ASCII keyboard; the TVT should display KIM and some memory location. From there on in, follow the KIM manual. Unless your ASCII keyboard is TTY oriented, it may not have a RUB OUT or DELETE character available — MOS says this is required to sense the baud rate. This isn't really a problem, as a question mark (?) (ASCII 0111 1111) seems to work just as well (on my system, at least, KIM seems to be happy with it). The baud rate may be adjusted with the SAB-1 clock adjust for 300 baud (4800 Hz) which is quite fast writing on the TVT, or you may experiment running it faster (it works) by running it up until KIM won't accept it. Remember: to let KIM find the new rate after an adjustment, you must hit RS, then RUB OUT or ? on the ASCII keyboard.

SWTP 6800 Interfacing

A 2N2222 transistor and 3 resistors are required to interface into the 6800 MP-C board. The SAB-1 clock should be adjusted with a counter to match the baud rate you have selected on the MP-C board. If a counter is not available, the twist and try method may be used, although it is a bit time consuming.

RS-232

RS-232 interfacing using two MCT-2 Optoisolators is shown for anyone desiring to access a serial port on their IMSAI or MITS systems.

Bay Area TVT

The jumper connections on J1 for the Bay Area TVT are shown so that a male-to-male 16 pin DIP connector cable may be used between the TVT and the SAB-1. A source for the TVT is shown in the parts lists.

A Word About UARTS

These nifty little serial-to-parallel data converters are just that, and more. It is worthwhile to take the time to read through a specification sheet; you'll get a world of information. The SAB-1 uses the UART in an asynchronous, unconditional mode; that is, if it sees the proper start and stop bits on the serial data, it dumps it in parallel format, and *vice versa*. There are probably a dozen other different modes in which it may be ordered, including synchronous with checks for framing error, parity, etc. So read the sheets, you may want to aim for that 30 kilobaud the manufacturer's specifications call out!

Kits

The SAB-1 is available as a complete kit, including sockets for all ICs @ $24.95 postpaid, from:

RGS Electronics
3650 Charles St. (Suite K)
Santa Clara CA 95050

The Bay Area TVT is available as a complete kit or bare board. It has ASCII keyboard input and direct video output and requires 5 V @ 1.25 A and -12 V @ 40 mA. Complete kit is $120 or bare board $35 plus postage and handling charge of $1, from:

BYTE SHOP #2
3400 W. El Camino Real
Santa Clara CA 95051

■

*Higher or lower baud rates may be obtained by changing C1, the 0.01 uF polystyrene; but since the UART is being operated in an asynchronous, unconditional mode, rates above about 2K Baud are not recommended because of possible data loss.

Letters to the Editor

from page 90

ments that would enable an EDP Center to gather information and then give the small businessman a daily readout of the status of his expected profit and loss. I noticed in today's LA Times a firm offering a similar service. They call themselves PAYFONE. Daily figures are read in by remote terminal and PAYFONE completes the data processing.

I am looking forward to the first issue of *Kilobaud*. I am sure it will be a success.

Anthony J. Oreb, Jr.
3700 Dean Dr., Apt. 2607
Ventura CA 93003

It appears that the majority of people are interested in stand alone systems for their small businesses, rather than something like the PAYFONE you mention. That "EDP Center" will probably be sitting on a desk in a small businessman's office, Tony. And, we'll be sure and keep you up to date on new developments in this area.
– John.

The Personal Touch

I am a fan of both Wayne Green and John Craig and would like to contribute toward your success with *Kilobaud*. Thus please allow me to take advantage of the $25 for a 3 year subscription to *Kilobaud* as stated by John in the October issue of *Doctor Dobb's Journal* and referred to by you in your December *73* editorial. My check is included herewith.

I like the personal touch you provide in your editorials — telling it like you see it — and hope you continue this policy in *Kilobaud*. I've found the computer articles in *73* consistently understandable without too much effort; this is more than I can say for *Byte*.

I'm not a ham and subscribe to *73* solely for the *I/O* articles but can't help occasionally reading a ham article and am finding myself attracted to that branch of electronics hobbying due to the appeal of your excellent magazine.

Please be sure that I get the first issue of *Kilobaud!*

Terif C. Young
Shepherdstown WV 25443

Definitely plan to keep Kilobaud *both personal and understandable.*
– John.



TOUCH TONE GENERATOR BY MOSTEK. MK5086N produces the dual-tone multi-frequency telephone dialing signals as used in TT phones and auto patches. Uses inexpensive crystal, 1 resistor and 1 capacitor. Both tones are internally mixed and buffered to a single output - simple! Two additional output switches can control timers, transmitter, mute receiver, enable audio amp, etc. Uses our Chomerics keyboard. Comes in 16 pin plastic DIP.
MK5086N.....$8.95...Crystal for MK5086N..... $1.90
Specs for MK5086N 80¢.
Kit of parts including etched and drilled P.C. board and one of our Chomerics keyboards.....$19.95

MC14412 UNIVERSAL MODEM CHIP
MC14412 contains a complete FSK modulator and de-modulator compatible with foreign and USA communications. (0-600 BPS)
FEATURES:
- .On chip crystal oscillator
- .Echo suppressor disable tone generator
- .Originate and answer modes
- .Simplex, half-duplex, and full duplex operation
- .On chip sine wave
- .Modem self test mode
- .Selectable data rates: 0-200, 0-300, 0-600
- .Single supply
 VDD=4.75 to 15VDC - FL suffix
 VDD=4.75 to 6 VDC - VL suffix

TYPICAL APPLICATIONS:
- .Stand alone - low speed modems
- .Built - in low speed modems
- .Remote terminals, accoustic couplers

MC14412FL..... $28.99
MC14412VL..... $21.74
6 pages of data..... .60
Crystal for the above.....$4.95

MC14411 BIT RATE GENERATOR.
Single chip for generating selectable frequencies for equipment in data communications such as TTY, printers, CRT s or microprocessors. Generates 14 different standard bit rates which are multiplied under external control to 1X, 8X, 16X or 64X initial value. Operates from single +5 volt supply. MC14411..... $11.98
4 pages of data..... .40
Crystal for the above..... $4.95

REMOTE CONTROL TRANSMITTER. MC14422P is a 22 channel ultra-sonic remote control transmitter I.C. CMOS uses little power and only a few external passive components. Applications include TV receivers, security controls, toys, industrial controls and locks. 16 pin DIP plastic pkg.
MC14422P..... with specs.....$11.10

PRECISION REFERENCE AMP
LH0070-1H provides a precise 10.0 volts for use in BCD A to D converters or meter calibrators. Typical initial accuracy is .3% (± .03V). Comes in TO-5 can.
LH0070-1H.....with specs.....$5.35

4 DIGIT COUNTER. MM74C926 is a 4 digit counter with 7 segment output. Carry output for cascading and internal display select allows outputting of counter or set of internal latches. 3 to 6V operation. Great for clocks, event and frequency counters.
MM74C926 - with spec sheet.....$12.00

3 DECADE (BCD) COUNTER CHIP
MC14553BCP consists of 3 negative edge triggered synchronous counters, 3 quad latches and self scan multiplexed , TTL compatible outputs.
MC14553BCP.....$8.72
Spec sheets.....$.60

TELETYPE CODE CONVERSION CHIP
MM52208L converts 5 level Baudot into 8 level ASCII. Use this chip to make your old TTY talk to your new computer.
MM52208L..... $18.00
Specs for the above..... .30

MOS TIME BASE KIT.
Only 1" X 1.5". Input 5 to 15 VDC, output is 60HZ square wave for portable or mobile clocks. PC board is drilled! MTBK-60HZ..... $5.88



HIGH POWER TRIAC.
Stud mount triac made by ECC. 200V, 25A. Part # Q2025D is perfect for lighting, motor control, heater control, solid state relays, etc. Q2025D..... $2.50

MINIATURE SCR. MCR106-4 is a 200V, 4A SCR in the tiny flat power pack. Only .27" wide X .13" thick (TO-92 case). Buy this one at OEM quantity prices!!!!
MCR106-4.....75¢, 10/$6.00

MM55106 PLL FREQUENCY SYNTHESISER
18 pin DIP package IC contains phase locked loop circuits useful for frequency synthesiser application, especially those in or near the CB band. Single supply operation; CMOS technology, binary channel select; programmable divider.
MM55106N.....$9.00. Specs.....40¢

MOTOR SPEED CONTROL SYSTEM.
uA7391 monolithic I.C. provides all functional blocks required for precision closed loop motor speed control. Use for 1% control accuracy on tape decks, industrial controls, etc.....$4.95.....Specs .60

DATA BOOKS BY NATIONAL SEMICONDUCTOR
DIGITAL. Covers TTL, DTL, Tri-State, etc. $3.95
LINEAR. Covers amplifiers, pre-amps, op-amps, .. $3.95
LINEAR APPLICATIONS. Dozens of application notes and technical briefs covering the use of op-amps, regulators, phase locked loops and audio amps..... Vol 1 $3.25
CMOS Gates, Flip Flops, registers, functional blocks $3
VOLTAGE REGULATORS. A must for anyone making a power supply. Complete theory including transformers, filters, heat sinks, regulators, etc..... $3.00
MEMORY. Information on MOS and Bipolar memories RAMS, ROMS, PROMS and decoders/encoders.. ... $3.95
INTERFACE. Covers peripheral drivers, level translators, line driver/receivers, memory and clock drivers, sense amps display driver and opto-couplers..... $3.95
(Outside U.S., add postage for 1.5lbs)

SPECIAL FUNCTIONS DATA BOOK contains detailed information for specifying and applying special amplifiers, buffers, clock drivers, analog switches and D/A-A/D converter products.....$3.25
AUDIO HANDBOOK contains detailed discussions, including complete design particulars, covering many areas of audio with real world design examples...$3.25

AMPL'ANNY Says

PROJECT OFF TO A SHAKY START?
GET OFF ON THE RIGHT FOOT
HEAD FOR THE PROS AT TRI-TEK

HORIZONTAL OUTPUT TRANSISTOR.
G.E. D56W1 is a silicon NPN high voltage power transistor designed for color and black/white TV horizontal deflection circuits.
ICEV @ 1400V = .5mA!!
VCE (SUS) = 600V minimum.
TO-3 POWER PACKAGE
D56W1 ----------Save on this one!! ----------$2.55
Specs for above..... .40

INCANDESCENT LIGHT DELAY.
Small module designed to fit directly behind your wall switch-plate. Turn switch off and "LITE-OFF" keeps light at half pawer for 15 seconds before turning off, allowing you to get from where you are to where you ain't with out breaking a leg. Up to 500W!!
LITE-OFF Model 100 w/instructions.....$2.15

MIDGET PUSH BUTTON SWITCH (CHEAP)
Flat shaped plastic body push button DPST-NO momentary switch. 1/4" bushing mount. Body only 1/4"X1/2" X 3/4" long. CPB-0201P.....3/$1.00, 10/$3.00

SOLID STATE RELAY.
Teledyne P/N 601-1010QQ is a heavy duty solid state relay module operating up to 10A at up to 250VDC. All brand new modules!! Still in original factory package.
1010QQ.....$6.88

8 AMP DARLINGTON
MJ1000 is a silicon NPN darlington in TO-3 case including a damping diode across emitter and collector.
VCEO=60V, IC max = 8 AMP
MJ1000.....99¢, Specs for MJ1000.....20¢

5 DECADE COUNTER
MC14534BCP is a 5 decade real time counter with multiplexed BCD outputs. Can be cascaded for longer counts. Typically 5MHZ operation at 15 Volts.
CMOS structure for low pawer consumption.
MC14534BCP.....$11.25

10 AMP VOLTAGE REGULATOR
MPC1000 is a 10 Amp positive voltage regulator adjustable from 2 to 35 VDC. 0.1% line and load regulation with 0.005% per C temperature stability. Can be fold-back limited. Here is high current, high pawer with minimum bother.
MPC-1000.....$16.85
Specs for above..... 60¢

NEW NATIONAL BOOK---LINEAR APPLICATIONS VOL II
Takes up where Vol I left you--All the latest linear devices. Along with Vol I you have a great source of application data on the most widely used devices as well as new types just appearing.....$3.25

INTRODUCTION TO MICRO COMPUTERS
New book from OSBORNE.
The first edition of this classic was a huge success. Now, due to the growth of information on the subject Osborne has expanded the work into 2 volumes. Vol I covers basic concepts, Vol II discusses real world micro computers.
IMC-002 Vol I.....$8.00
IMC-002 Vol II..... $8.00

'NOTHER NEW BOOK FROM OSBORNE.
"8080 PROGRAMMING FOR LOGIC DESIGN" explains how an assembly language program within a microcomputer system can replace combinatorial logic ---- for logic designers, programmers or anyone who is interested in real and powerful applications of the ubiquitous 8080.
PLD-4001..... $8.00

TRI-TEK, INC.
6522 NORTH 43RD AVENUE,
GLENDALE, ARIZONA 85301
PHONE 602 - 931-6949

We pay shipping on all orders over $10 US, $15 foreign in US funds. Orders under $10, please add $1 handling. Please add insurance. Master Charge and Bank America cards welcome, ($20 minimum) Telephone orders may be placed 11AM to 5PM daily, Mon thru Fri. Call 602-931-4528. Check reader service card or send stamp for our latest flyers packed with new and surplus electronic components.



Front panel of Logic Analyzer Box (LAB), model LAB-8.

R.A. Walker
T.H. Lincoln
A.H. McDonough
6441 Hughes Dr.
Huntington Beach CA 92647

Nobody Knows the Troubles I've Seen

This article describes a compact low-cost tool for use in the analysis of logic circuits. The device connects to as many as 16 logic lines. An internal or external trigger is applied, and if a signal is present on any of the logic lines when the trigger occurs one (or more) electronic latch is set and a corresponding LED is illuminated.

The spread of digital technology has been so great in recent years that the manufacturers of test instrumentation have, at last, taken notice. Both of the premiere manufacturers of electronic testing devices have, in the last year, introduced test devices aimed at the digital computer/logic field. Even more encouraging, these devices have shown some recognition of the needs and problems of the poor soul faced with a maze of high-speed, nonrecurring pulses. The "logic analyzer" has proven to be an invaluable

The authors of this article are a good example of how some of the professional digital designers are going to make some significant contributions to our hobby. They are, incidentally, in the same boat we're all in; that is, having to make sure each buck counts when it comes time to buy something to sustain their hobby (home computers). Therefore, you'll notice they have a reasonable price for those of you interested in their assembled unit. We're going to be hearing more from the guys at "L" Electronics . . . as a matter of fact, they have a couple of expansion/enhancement goodies for the LAB coming up in a future article (they use a $7,000 HP Logic Analyzer at work . . . maybe they're going to try building something like that for the hobbyist). — John.

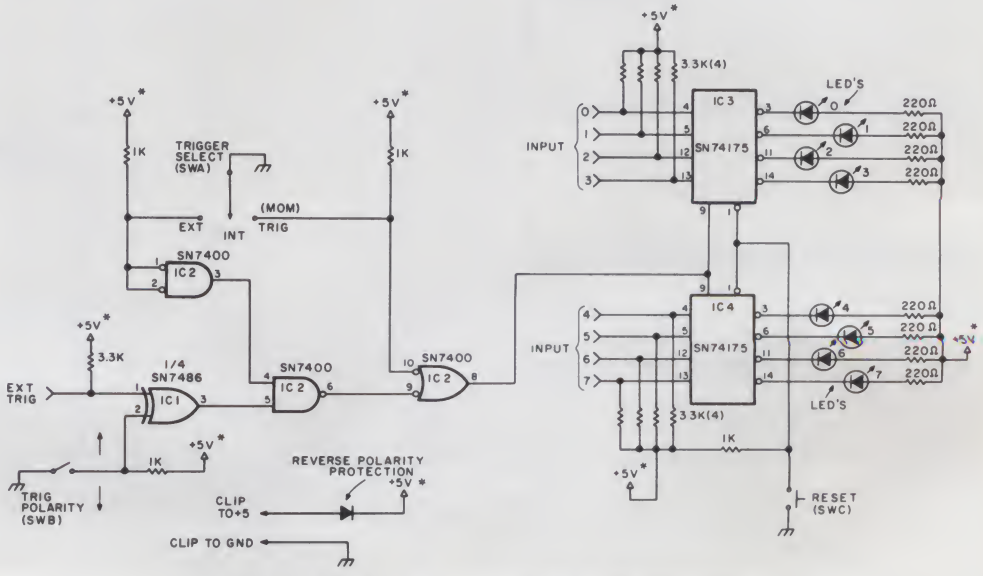
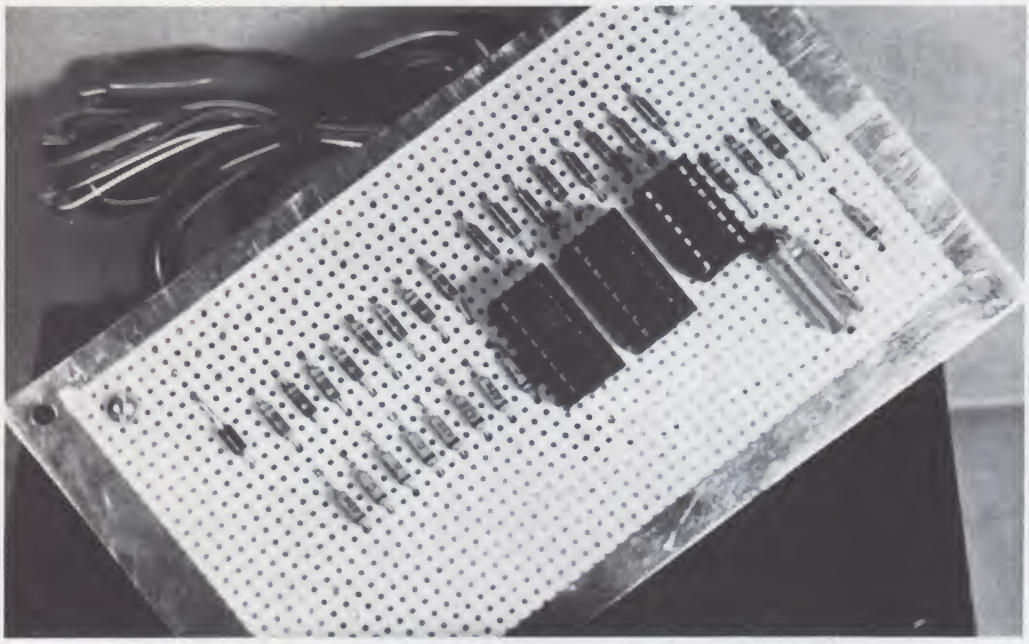


Fig. 1. LAB-8 logic circuit diagram (8-bit version).



Prototype circuit board assembly for Logic Analyzer model LAB-8.

tool in the development and debugging of digital systems but, at prices ranging around $7000, far beyond the reach of all but the richest computer hobbyists.

Most hobbyists try to "make-do" with an oscilloscope. However, it doesn't take long to discover that some tests are difficult, if not impossible, to perform even with a very good scope. There are many instances when even a small logic analyzer is better than the best of scopes and a good deal easier to use and interpret.

For the hobbyist, or the professional, the compact and inexpensive logic analyzer shown in Photo 1, can be a valuable tool. The logic analyzer is connected to up to 16 digital lines and to the line supplying the trigger pulse. When a trigger pulse occurs, the analyzer captures the state of the lines at that instant and, if the line was "true," illuminates a corresponding LED. This display is retained until a new trigger pulse occurs or until the RESET button is depressed. A manual trigger switch allows the user to read out the static state of the lines under examination at any time.

Theory of Operation

The logic analyzer circuitry (shown in Fig. 1) is relatively simple and yet allows considerable flexibility of operation. The RESET switch (SWC) returns all the latches in I3 and I4 to the OFF state, extinguishing all the LED's. The TRIGGER select switch (SWA) is normally open in the center position. It toggles to select external triggering (EXT) and makes momentary contact in the internal trigger position (INT). After pressing RESET, if SWA is pressed to INT, the pulse generated by the momentary contact passes through OR gate I2 and causes the latches in I3 and I4 to capture the current state of the input lines.

If switch SWA is set to EXT, the gating is arranged to let external trigger pulses enable the latches. THE TRIGGER POLARITY switch (SWB) connects the exclusive OR (I1) as either an inverting or noninverting buffer, allowing external triggering with either positive- or negative-going pulses. The trigger pulse, whether externally or manually generated, is applied to the clock inputs of the SN74175s (I3 and I4). This latches the data present during the positive edge of the trigger pulse.

The new data is loaded over any existing data each time the unit is triggered so it is not necessary to reset between tests. However, use of the manual reset between tests eliminates possible ambiguity. As an example, if a test indicates some error in a circuit you may wish to replace an IC chip and repeat the test. If the data display does not change, it could either be that the replacement did not affect the data or that the replacement has somehow disabled the trigger pulse and new data was never loaded. Use of the manual reset between tests eliminates this uncertainty.

Construction

Layout of the components is not critical. The prototype unit, shown in Photo 1 was wire-wrapped and built into a small plastic box with a metal front panel. The circuit assembly is shown in Photo 2. A printed circuit board was developed for easier assembly of subsequent units.

Increasing the value of the 220 Ohm resistors will reduce

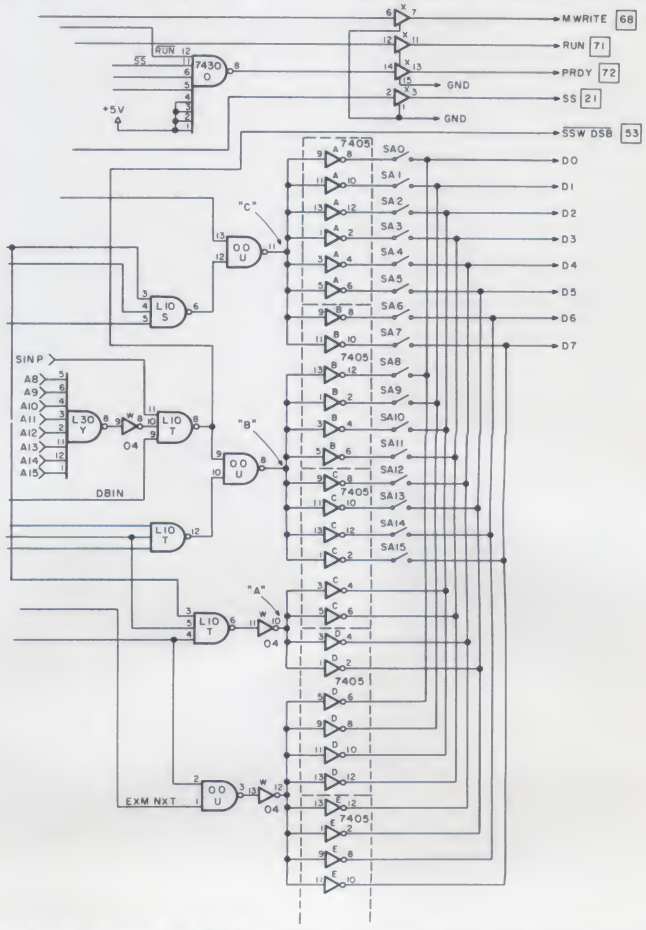


Fig. 2. Altair 8800 front panel control circuitry (partial).

the current to the LEDs and extend their lives. The use of alligator clips is suggested for power and ground to permit stealing power from the circuit under test.

Either kits or assembled and tested units are now available with the printed circuit board.

Applications

The use of the logic analyzer can be illustrated by considering some fictitious troubleshooting operations on a currently popular piece of equipment, the Altair 8800. The intent of these examples is not to provide detailed troubleshooting procedures, but only to illustrate the usefulness of the logic analyzer.

First, we assume there is some trouble in the front panel control circuitry. This trouble has shown itself to be a failure of the EXAMINE operation. The EXAMINE function allows manual selection of a memory address by front panel switches and display of the contents of that address. The relevant circuit is shown in Fig. 2. The address to be displayed is set on switches SA0 through SA15. When the EXAMINE switch is depressed, three operations take place automatically at very high speed.

First, octal 303 is placed on lines D0 through D7, regardless of switch settings. Next, the low order byte of the address to be examined is placed on the same line, and last, the high order byte is pulsed on those lines. In summary, the lines D0 through D7 carry three different bytes, occurring sequentially and automatically when executing a single EXAMINE operation.

The standard test equipment available in most home labs is of little help in studying such a function. A voltmeter is of no help and a scope is little better unless you are fortunate enough to have a memory scope available. Even with a memory scope you can look at only one or two lines at a time. The logic analyzer, however, can supply you with a lot of information at one time.

Connect the logic analyzer inputs to lines D0 through D7 of the 8800. Connect the trigger input to ICW pin 10 (Point "A" in Fig. 2), of the 8800. This point will go high at load time in order to drive lines D2, D3, D4 and D5 to ground. Set the analyzer TRIGGER POLARITY switch (SWB) to ↑, which means you expect a positive going transition at trigger time. Press RESET to clear any old data stored in the latches. Set the TRIGGER select switch (SWA) to EXT and press the EXAMINE switch on the 8800. The logic analyzer should now contain the octal number 303.

If octal 303 is not displayed, you have determined the area in which to start looking for your first problem. A completely blank display (octal 000) may indicate a failure to trigger. A logic probe, such as the beeper described on page 106 of the August '76 issue of *73 Magazine*, will spot this problem.

Assuming you did get an octal 303 on that first test, the next step is to move the trigger input to ICU pin 8 (Point "B"). All switches should remain unchanged except to press RESET and clear the display. The logic inputs should remain connected to D0 through D7.

Press the 8800 EXAMINE switch again and note the logic analyzer display. It should indicate the low byte address. Using ICU pin 11 (Point "C") for the trigger point will allow examination of the high order address sent to the CPU.

For a second example, consider a different part of the 8800, the static memory board circuitry shown in Fig. 3. We assume that you have tried to write a word into memory and find you are unable to read that word back. It is not immediately possible to tell if the problem is in writing to, or reading from, memory.

Start by connecting the data lines from the logic analyzer to ICH pins 10, 2, 6, 4 (Point "A") and ICJ pins 6, 4, 2 and 10 (Point "B"), corresponding to DI0 through DI7 respectively. Trigger from ICG pin 11 (Point "C"). If you have the word previously stored, move the logic

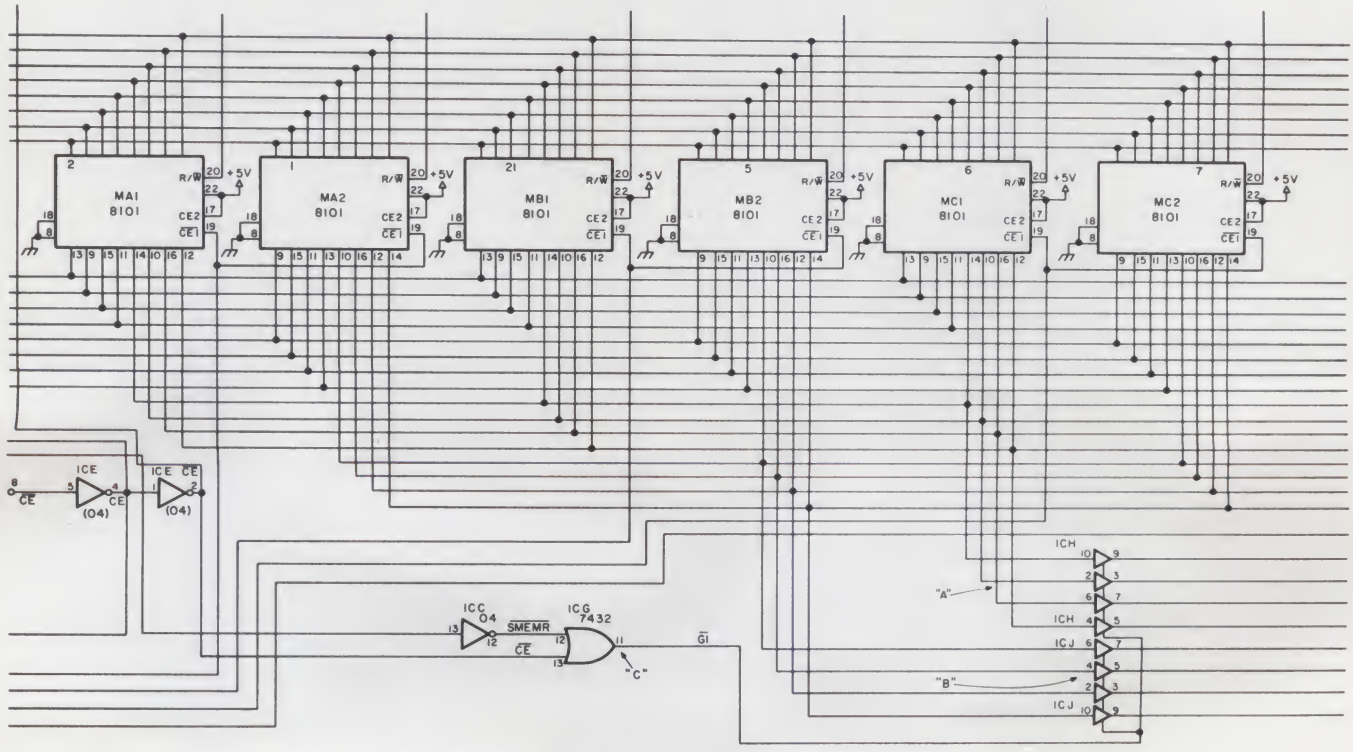


Fig. 3. Altair 8800 1K static memory board (partial).

analyzer inputs to the data lines DI0 through DI7 and test again. If you no longer have the data word, you have probably isolated the problem to ICH or ICJ.

As you can see the problem reduces to one of determining where you want to look and when you want to sample. Sample time determines the trigger points.

Summary

Both of the previous examples illustrate the testing of parallel data lines. This is by no means the only possible application. It may be necessary to examine a single logic line at several points. The logic analyzer inputs can be connected to the inputs and outputs of several logic elements, either inverting or noninverting, to determine if a given signal has been successfully transmitted by each of the elements.

This should give some indication of the usefulness of this tool. A logic analyzer will not replace your scope but it will do very well some jobs that a scope will not do at all.

The logic analyzer can be expanded from eight to sixteen or more bits simply by adding SN74175s and the necessary LEDs. In some applications it may be desirable to trigger on a code word. This can be easily accomplished by adding a qualifier gate on the EXT TRIGGER input.

A kit is available for either the 8- or 16-bit models. The kit includes the plastic and metal box shown in Photo 1, a silk screened panel, a printed circuit board, and all required components. The 8-bit model is designated LAB-8, the 16-bit model LAB-16. The price of the LAB-8 is $55 and the price of the LAB-16 is $65. Both prices include postage within the United States. Order from "L" Electronics, 410 Bell Avenue, Santa Ana CA 92707. ■

IMSAI

I-8080 – Tabletop version of basic computer system . $539.00

EXP-22 – Twenty-two slot mother board, when ordered with basic system 46.80

Illinois residents please add sales tax. We will ship UPS prepaid. We honor BankAmericard and Master Charge. Send us $1.00 for catalog & $1.00 credit memo.

Quality Security Systems Computer Sales
3407 Chambord Lane
Hazelcrest IL 60429

MICROCOMPUTER PROGRAMMING COURSE

FREE description and outline of MODU-LEARN™ Home Study Course in Microcomputer Programming. Hundreds of pages of text with examples, problems and solutions. Prepared by professional design engineers using systematic software design techniques, **structured** program design, and **practical examples** from real microcomputer applications. Presented in a modular sequence of ten lessons oriented for the engineer, technician or hobbyist beginning to need programming skills. Includes background material on microcomputer architecture, hardware/software tradeoffs, and useful reference tables. Much of this information has been available only through costly seminars. Now you can study this complete course at home at your own pace for only $49.95. Send for FREE descriptive brochure now.

LOGICAL SERVICES INCORPORATED
711 Stierlin Rd
Mountain View, CA 94043
(415) 965-8365

IN NEW ENGLAND THERE ARE TWO

Computer Mart Stores

Featuring

TDL • SW Tech

Sphere • IMSAI

Digital Group

1097 Lexington Street
Waltham MA
617-899-4540

Route 3
Merrimack NH
603-424-2981

THE COMPUTER CORNER

White Plains Mall
200 Hamilton Ave.
White Plains, NY 10601
Phone: 914•WHY•DATA

Right off Bronx River Parkway. Plenty of parking.

"STAY ON THE BUS" with the largest collection of boards compatible with the Altair Bus (also IMSAI) in the greater NY area.

You've read about the Sol-20, now come up and see it. We carry Polymorphics, IMSAI, cassettes, etc., etc.

Ask us about beginners classes and information on clubs. We provide service.

THE COMPUTER CORNER

THE COMPUTER WORKSHOP

Microcomputers	Digital Group
Floppy Disks	IMSAI
Printers	Nat'l Mux.
Terminals	Oliver
Digital Tape Recorders	POLY
Special Interfaces	Seals
TV Monitors	SwTPC
Software	SPHERE

TDL
& others

Expert Help & Advice

(Kansas City Area)
6903 Blair Rd.
Kansas City MO 64152
tel. 816/741-5055

(Washington, D.C.)
5709 Frederick Ave.
Rockville MD 20852
tel. 301/468-0455

John Craig
Editor



Dr. John Kemeny

Structured BASIC

... A Negative View by Dr. Kemeny, the Author of BASIC

The Beginner's All-purpose Symbolic Instruction Code, more commonly referred to as BASIC, was developed in 1965 at Dartmouth College in New Hampshire. The development was supported by a grant from the National Science Foundation and was directed by Professors John G. Kemeny and Thomas E. Kurtz. They recognized the need for an easy-to-use computer language which would make it possible for the layman to gain the benefits from a computer *immediately*, without having to spend endless hours learning a language before being able to apply it. Needless to say, this accurately describes the results of their efforts and explains why BASIC is *the* language in widespread use within the hobby community today.

Dr. Kemeny, who is now president of Dartmouth, had no contact with the hobbyist movement prior to being approached by *Kilobaud* for this interview. He has some interesting thoughts to share with us regarding the development of the language, some of the criticisms which have been leveled at it lately, and where it is going in the future.

Kilobaud: Was BASIC developed as a result of a bet between you and Professor Kurtz?

Kemeny: If there was a bet, it was only whether a language developed at a school could ever compete with FORTRAN. I bet that it could, and Tom Kurtz had serious reservations on it. Nevertheless, we developed the language together. We both felt it would be worthwhile to develop a new language.

Let me set the scene. At that time FORTRAN was the only commonly used user-language. Several years after FORTRAN was developed, we felt that a more easily used language was desired, one of the considerations being that time-sharing has somewhat different requirements than did batch-processing systems.

Kilobaud: Did you ever think it would catch on the way it has and enjoy such popularity?

Kemeny: No, neither one of us ever dreamed it would become as popular as it has.

Kilobaud: You mentioned FORTRAN a moment ago ... would you discuss for a moment the relationship between FORTRAN and your BASIC?

Kemeny: We actually started developing the language by looking at FORTRAN and seeing what we liked in it and what we disliked in it. And of course, later on, FORTRAN incorporated some of the things we had put into BASIC. So, I think in many ways the difference between FORTRAN and BASIC is less than it originally was and each language has profited from the other language. I'll tell you what my two major concerns were when I developed BASIC. In the original FORTRAN, you had to learn an enormous amount before you even got started. You had to learn an awful lot of things before you wrote your first program. I thought this was not good for a beginning user. Therefore, we wrote BASIC in what we call "levels." We could get up to a given level, live there quite comfortably ... and then move onto the next level.

The second concern was that the time-sharing would be in a conversational mode. We felt that would enable us to simplify things for the user more than was possible at that time using FORTRAN. We were after such features as having line numbers and automatic editing of a line by retyping. These features have since been incorporated into other time sharing languages, but BASIC was the first to have them. Of course, it was the first to have an INPUT statement, because it only made sense in time sharing for the user to supply data.

Kilobaud: Would I be correct in assuming that you're aware of growth of the hobby movement and the use of BASIC in home computer systems the last couple of years?

Kemeny: Yes, I've heard a good deal about the hobby movement.

Kilobaud: As far as home systems are concerned, do you see any possibility of BASIC being replaced in the future with another language?

Kemeny: I don't see any need for that. Obviously, if somebody comes up with a better language it should be

used. But, what might be ideal for the hobby systems would be our original effort called, Baby BASIC. There is an enormous amount that can be done with minimal BASIC, and it can be implemented on a very small computer.

Kilobaud: Have you had any inputs, or a voice, with regard to the standardization efforts currently underway for BASIC?

Kemeny: Not personally, but Professor Kurtz and another Dartmouth professor Steven Garland have played a leading role (particularly Professor Garland) in the standardization of BASIC . . . and we're very happy with the way it is coming out.

Kilobaud: As far as you know, has the standardization committee received all of the inputs by now?

Kemeny: Yes, they have even made contact with an overseas group working on it and I think all the major problems have been resolved.

Kilobaud: From what I've heard there is not a BASIC in existence which meets the *minimum* proposed standards, because of the need for an OPTION state command (which allows the operator to specify whether an array starts at zero or one). There is no OPTION command in any BASIC around.

Kemeny: That may be. I wasn't aware of that fact. I did ask and I know we're going to have to make some minor changes in our BASIC to live up to the standard . . . but I was told they were quite minor.

Kilobaud: Recently there have been some articles in several publications which have been critical of BASIC with regard to the fact it doesn't lend itself to structured programming. What are your feelings on this matter?

Kemeny: Let me express a personal prejudice, which, incidentally, many of my colleagues don't agree with. I think some of the arguments for structured programming have been badly exaggerated. It reminds me of a branch of mathematics which becomes very well developed and the purists take over and impose upon that branch conditions of high purity. I'm not a great believer in that . . . I feel that for the majority of programmers such things as structured programming just get in the way. I think a relatively few safeguards, like those in ordinary programming, can achieve 95% of what structured programming can.

Kilobaud: At this particular point in time it really seems a waste to be sitting around discussing BASIC's shortcoming. What we really need, especially in the hobby community, is to get busy and develop some good applications software. That's where we're lacking right now.

Kemeny: Yes, I quite agree with that. BASIC has lent itself to tens of thousands of users writing endless numbers of very good applications programs and none of them seem to be handicapped by any particular feature of BASIC.

Kilobaud: Fine . . . I would have expected a response such as that from you, but it was nice hearing it anyway.

Well, Dr. Kemeny, I guess that should just about do it. Have you anything to pass along in summary?

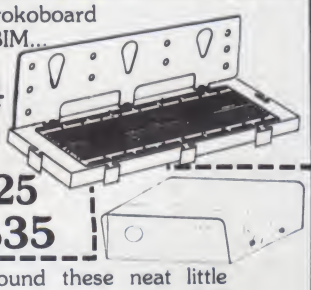
Kemeny: Yes, let me express one basic prejudice which is relevant to our discussion on structured programming. I think that it is terribly important that the computer experts, and though I consider myself one of them, I still have a prejudice that computer experts should not be allowed to interfere too much with the pleasure of the lay computer user and put in too many limitations as to what the lay user can do. ■

from **proko!!**
IMSAI 8080
$550



IMSAI's new I/O board $185
* 2 parallel ports * cassette interface
* 1 serial port * control port

The prokoboard from BIM...
$14



2/$25
3/$35

We found these neat little aluminum boxes and couldn't resist. They have a nice walnut finish. Great digital clock case! 2 for $3.00

NEW!
The Proko Paper-Tape Reader...$42.00

Check or Money Order only. Calif. Residents add 6% tax. All orders postpaid in the U.S. $10 Min. order. Prices subject to change without notice.

the proko electronics shoppe
439 marsh st.
san luis obispo, ca. 93401
805/544-5441

NEW! From Solid State Music

64 x 16 VIDEO BOARD (At last) Altair plug compatible display 32 x 16 or 64 x 16 switch selectable. Composite and parallel video ports, 8-91L02A memory for graphics and text, upper and lower case 2 x 3 grid for each location on graphics, with software. Kit.....$179.95

4Kx8 Static Memories
MB-1 MK-8 board, 1 usec 2102 or eq. PC Board $22 Kit.....$83

MB-2 Altair 8800 or IMSAI compatible Switched address and wait cycles. PC Board.....$25 Kit (1 usec) $112 Kit (91L02A .5 usec).....$132

MB-4 Improved MB-2 designed for 8K "piggy-back" without cutting traces. PC Board $30 Kit 4K .5 usec.....$137 Kit 8K .5 usec.....$209

MB-3 1702A's Eroms, Altair 8800 & Imsai 8080 compatible Switched address & wait cycles. 2K may be expanded to 4K. Kit less Proms.....$65 2K Kit.....$145 4K Kit.....$225

MB-6 8Kx8 Switched address and wait assignments. Memory protection is switchable for 256, 512, 1K, 2K, 4K and 8K. 91L02A .5 usec rams, Altair 8800 & IMSAI compatible. Kit.....$250 Assembled & tested.....$290

I/O Boards
I/O-2 I/O for 8800, 2 ports, committed pads for 3 more, other pads for EROMS UART, etc.
Kit.....$47.50 PC Board only.....$25

Misc
Altair compatable mother board. Room for 15 connectors 11" x 11½" (w/o connectors).....$45
With 15 connectors.....$115

Altair extender board (w/o connectors).....$9
With w/w connector.....$14

90 Day Guarantee on SSM Products Kits MB-2, MB-3 (2K OR 4K), MB-4, MB-6, IO-2 and mother board with connectors may be combined for a discount of 10% in quantities of 10 or more. *This supercedes* the flier of 13 Sept. 1976.

1702A* EROM	$10.00
1702A* 2 usec	8.00
*programming send hex list	5.00
AY5-1013 UART	$6.95
2513 prime spec. upper or lower case	11.00
8080A prime CPU	25.00
8212 prime latch buffer	4.00
8224 prime clock gen	5.00
8228 prime sys controller	8.90

MIKOS
419 Portofino Dr.
San Carlos, Ca. 94070



Please send for xistor, IC & kit list

For large orders please send money order or cashiers check to avoid delays in waiting for checks to clear.

Check or money order only. Calif. resident 6% tax. All orders postpaid in U.S. All devices tested prior to sale. Money back 30 day guarantee. Sorry we can not accept returned IC's that have been soldered to. $10 min. order. Prices subject to change without notice.

Thinly disguised affiliates of KO Electronics and Surplus, S.L.O., CA 93401

Glossary

This glossary was researched and compiled by Doug Hogg of Santa Barbara CA with assistance from John Goettelmann of Pt. Pleasant NJ. To try to make it as useful as possible we've read each article with an eye to terms which we felt might be confusing to the beginner. If you feel we've missed a term that should have been defined, please drop me a note asap, and we'll try to get it to press for the next issue. — John.

ALGORITHM: A rule, method, or procedure used for solving particular problem or class of problems.

Normally, in programming, the word is used to describe a simple, yet powerful, rule which may be applied to solve a commonly encountered problem, one of the two basic ingredients of any program (the other being data). The algorithm specifies the rules by which the data is to be processed in order to meet the objective of the program.

Example:

Problem Statement	*Algorithm*	*Result*
Add the numbers in Boxes A and B together and write the result in Box A.	1. Take the contents of Box A and add it to the contents of Box B.	A 600
A 500		B 100*
B 100	2. Place the sum in Box A.	

*Box B is unchanged since no instructions were given to *change* it.

AND GATE: A circuit element whose output will be a logical one when, and only when, all of the inputs are in a logical one state. The truth table for a two input AND gate is shown below.

Inputs A	Inputs B	Output
0	0	0
1	0	0
0	1	0
1	1	1

The symbol for the AND gate is:



Also see logic.

ARCHITECTURE: A loosely defined term which refers to the interrelationships between the principal parts of a computer system and the methods and paths by which data can be made to flow within the system.

Aspects of a computer system which are usually investigated when considering the architecture of one system over another are:

The number of registers available and how they can be utilized

The number of stacks available and how they can be utilized

The instruction set with regard to the control of input/output operations

The hardware interrupt structure

The number and kinds of data paths into memory

The number and kinds of data paths into the CPU

Architectures are generally classified into one of two major categories:

Central processor oriented

Memory oriented

Each category has its own set of advantages and disadvantages, depending on the intended use of the system and permissible costs. The memory-oriented architecture is by far the most flexible, since it permits both the CPU and the various input/output devices to access memory freely and independently, if desired. The CPU-oriented architecture is frequently less expensive, but suffers in performance because the CPU has to move, or at least control, all of the data moving into and out of memory.

ARGUMENT: In programming, it is frequently desirable to design a subroutine to acquire a sequence of one or more data items from the part of the program which called it. The subroutine uses these data items, applying some algorithm, to accomplish a specific result. *The data items passed from the calling to the called parts of a program are referred to as arguments.*

As one moves from simple assembly language programs to higher level language programs, topics relating to arguments and how they are actually passed between the various program segments become quite complex.

ASCII: Acronym for American Standard Code for Information Interchange. This is a code which assigns a seven bit binary number for each letter of the alphabet, numbers and punctuation marks. In addition, certain machine codes (carriage return, line feed, end of transmission, tab, etc.) are also assigned codes. This is the standard coding used in TV Typewriters, all of the hobbyist video driver systems and the newer TTYs (as opposed to the older, Baudot code machines).

ASYNCHRONOUS OPERATION of a system is at a rate not constant in frequency and phase with respect to a master clock. The input of a TTY to a computer is an asynchronous operation since the inputs occur at random times determined by the system operator, not the computer clock.

BAUD: A data transmission rate of one bit per second. For example, the standard TTY speed is 110 baud, or 110 bits per second.

BAUDOT: A code, similar in concept to ASCII, assigning numbers to letters of the alphabet, numbers and punctuation marks. This code uses only 5 binary bits (32 possible combinations) and so does not have a unique code assignment for each character. There are two codes for case shifting (letters and figures) which allow a total of 60 different characters to be represented in this code. Thus to send "a15B" the following characters would be sent: *letters, A, figures, 1, 5, letters, B.*
See also ASCII.

BCD: Binary Coded Decimal. A method of representing decimal digits in the form of 4 bit binary words as shown below:

Decimal	BCD	Binary
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111
8	1000	1000
9	1001	1001
10	1 0000	1010
11	1 0001	1011
12	1 0010	1100
13	1 0011	1101
14	1 0100	1110
15	1 0101	1111

BCD may be converted to decimal by converting each group of 4 bits to a decimal digit. Note that this does not work for straight binary.

BINARY: The base two numbering system. Each digit may be either a 1 or a 0. In decimal (base 10) each digit to the left increases in value by a factor of 10. In binary each digit to the left increases by a factor of 2. The table shows the binary equivalents of decimal 0 to 7.

Decimal	Binary
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111

Also see BCD, octal, hexadecimal.

BIPOLAR refers to the use of conventional transistors (NPN or PNP) as opposed the field effect transistors (FETs). The distinction arises since the bipolar transistor is constructed from semiconductor material with sections of both N-type and P-type doping while each MOS FET is made of material with only one type of doping.

BIT is one *bi*nary digi*t*. For instance, the binary number 11001 has 4 bits.

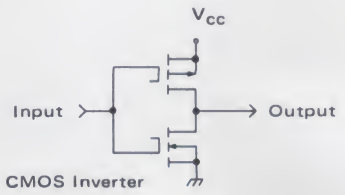
BUFFER (DRIVER) is a circuit used to isolate a load from the driving source. For instance, a buffer is usually used between a microprocessor chip and the memory address lines to boost the small drive current from the microprocessor IC (typically less than 2 mA) to a higher value suitable for driving a large number of devices simultaneously (typically a maximum of 50 mA). The most common types of IC buffers are the 8T97 and 8097 which have 6 buffers per chip, require .25 mA at the input and will drive up to 50 mA at standard TTL voltage levels.

BUFFER STORAGE AREA: A temporary storage area for data. This may be a single word of storage (such as in an output port latch holding data until the output device has accepted it). A buffer area in main memory can range from a single location to a large block.

BUS: A group of connections which carry information to the various parts of the computer. As an example, the data bus lines go to the CPU memory and input-output devices. A typical microprocessor bus system consists of parallel lines which carry the processor status signals, memory address, data and control signals to all of the circuit boards. In such a bus system, any card may be plugged into any position since each pin on each connector carries the same signal.

BYTE: A unit of data which consists of eight binary digits.

CMOS: Complementary MOS. This refers to a family of integrated circuits whose output structure consists of an N-type MOSFET and a P-type MOSFET in series. The term complementary is used since the N and P-type MOSFETs are complements of each other.



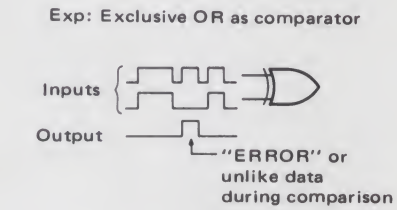
CPU: Abbreviation for Central Processing Unit. This is the heart of the computer which actually carries out the instructions contained in memory. The 8080, a typical small CPU, has several internal registers for holding temporary results and addresses, a stack pointer which contains information used in subroutine calls and returns, a program counter which contains the address of the next instruction to be executed and an arithmetic logic unit (ALU) which performs the mathematical and logical operations. A complete computer consists of the CPU, memory and input-output devices (I/O).
Also see MPU.

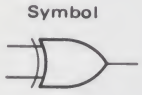
DIP: Abbreviation for Dual Inline Package. This is a case type commonly used for integrated circuits. The package has two parallel rows of pins on either side of the package spaced .100" apart. Packages presently available have either 8, 14, 16, 18, 20, 22, 24, 28 or 40 pins.

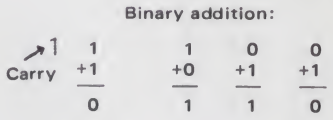
EROM: Eraseable Read Only Memory. This is a special type of read only memory which can be programmed electrically. The unique feature is is that it retains data even with the power disconnected but can be erased by exposure to short wavelength ultraviolet light (sunlight is not sufficient). The device may be reprogrammed many times. Common types are the 1702A (256 8 bit words), the 5204 (512 8 bit words) and the 8708 (1024 8 bit words).
Also see ROM, PROM and RAM.

EXCLUSIVE OR: The exclusive OR gate will provide a high output only with unlike inputs (i.e., one input high and the other low). The Exclusive OR is commonly used as a comparator and can also be used to perform simple binary addition. The truth table and logic symbol are shown below.

Inputs A	Inputs B	Output
0	0	0
1	0	1
0	1	1
1	1	0







Also see logic.

FLAG: A single bit used to indicate the result of a test. Typical microprocessors have flags for such functions as zero test, positive test, and carry. For instance, if the contents of a register is zero, the zero flag will be set. If a register contains 001, then only the positive flag will be set (the most significant bit is the sign – "0" = positive, "1" = negative). Flags are generally used as the basis for conditional jump decisions (Jump if Zero, etc.)

HEXADECIMAL: A method of representing numbers using base 16, as shown below.

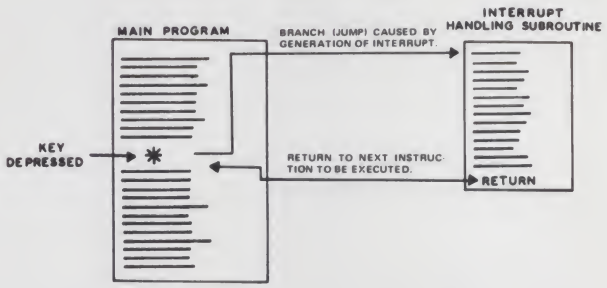
Decimal	Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F
16	10

Binary can be converted to hexadecimal by summing each group of four binary digits. For example:

Binary	1011 0110
Hex	B 6

Also see BCD, binary, octal.

INTERRUPT usually refers to a hardware signal used to represent the occurrence of a real time event within a computer system. The occurrence of this event (and the generation of an interrupt signal) forces the computer to immediately stop whatever it is doing and take appropriate action in response to the event.



Example: Suppose your home computer is being used to constantly monitor the environment. It's checking the temperature sensors in the lawn to see if the lawn needs water, the time of day to adjust the solar panels on the roof. It's also monitoring the security and fire systems, just to name a few.

If you desire to communicate with your computer, it's going to be done on the basis of "interrupting" these other functions. When you hit a key on the keyboard (to request a particular function) an interrupt is generated. The processor completes the instruction it is currently executing . . . and then begins executing an *Interrupt Handling Subroutine* which will determine which key on the keyboard was depressed and exactly what action needs to be taken. At the completion of the subroutine, control is returned to the main program (the one which was originally interrupted).

INTERRUPT LINKAGE refers to the technique which causes the computer to switch to (and then return from) the "interrupt handling" portions of the program as various interrupts occur.

KILOBAUD: A data transmission rate of 1000 baud (one thousand bits per second).

KIM: A self contained microprocessor board produced by MOS Technology Inc. The one board unit has a 6502 CPU, 2048 words of ROM, 128 words of RAM, a keyboard, LED display, a TTY interface and an audio cassette interface.

LOCATION refers to a position or address in memory (for example, making reference to the data in location 377).

LOGIC: In electronics, certain circuits are used to perform logical functions. The functions are AND, NAND, OR, NOR and Exclusive OR. The output of these circuits is dependent on the state (1 or 0) of the inputs.

MEMORY PAGE: A section of memory, typically 256 words. This arises from the fact an 8 bit computer handles memory addresses in 8 bit bytes. One byte can address 256 locations so most 8 bit microprocessors use a total of 2 bytes to give one 16 bit word capable of addressing 65,536 (2^{16}) locations. The upper 8 bits are referred to as the page number. Thus the address in octal page form 012 125 is location 125 (octal) on page 012 (octal).

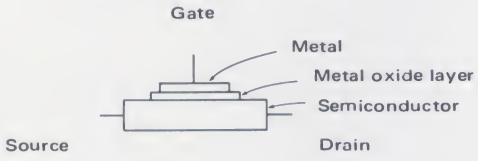
MEMORY PROTECT refers to the ability to selectively disable the write line on a memory segment. In this way the information in the memories can be protected from accidental destruction from inadvertantly writing in new information. On some memories this feature is set manually and on some units the memory protect can be set or reset under control of the operating program. This system does not protect against loss of the memory data due to interruption of the power supply.

MNEMONICS: A technique of improving the memory. In programming the symbolic codes used for instructions, addresses and data in assembly-language are referred to as mnemonics. These symbols (for instructions, in particular) are certainly easier for the programmer to remember than the octal or hexadecimal equivalent in machine-language.

MOS: Acronym for Metal Oxide Semiconductor. This refers to the three layers used in forming the gate structure of a field effect transistor (FET).



Junction FET. The gate is P-type semiconductor diffused into the bulk material.



MOS FET. The metal gate is separated from the main body of the FET by an insulating layer of metal oxide.

The bulk material may be either N-type or P-type. These are called NMOS and PMOS. Because this is a simple process, many of the higher density integrated circuits (particularly memories) use this process. These integrated circuits are referred to as MOS ICs.

MPU: Abbreviation for *micro*processor *u*nit. Refers to a Central Processing Unit (CPU) implemented with an integrated circuit microprocessor.
Also see CPU.

NAND GATE: A circuit element whose output is the negation (or complement) of the logical AND function. The output is logic level 0 only if all the inputs are at logic level 1. The truth table and symbol for a 2 input NAND gate is shown below.

NOR GATE: A circuit element whose output is the negation (or complement) of the logical OR function. The output is a logical zero if any of the inputs are at logical one. The truth table and symbol for a 2 input NOR gate is shown below.

Inputs		Output
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

OBJECT Format or Program: See **SOURCE PROGRAM.**

OCTAL: A method of representing numbers using base 8. The first 12 numbers and their decimal equivalents are:

Decimal	Octal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	10
9	11
10	12
11	13
12	14

Binary can be converted to octal by taking groups of three binary digits and summing the values. For instance,

Binary	101 010
Octal	5 2

Also see BCD, Binary, and Hexadecimal.

ON-BOARD REGULATION refers to the practice of placing small voltage regulators on each circuit board in a system rather than having one large power supply for the entire system. Altair-compatible circuits use on-board regulation. Advantages include the ability to expand the system gradually, the isolation of electrically noisy circuits, and no noise pickup on the power supply wiring. The disadvantages include the necessity for a large number of voltage regulators and the placing of heat sources on the cards.

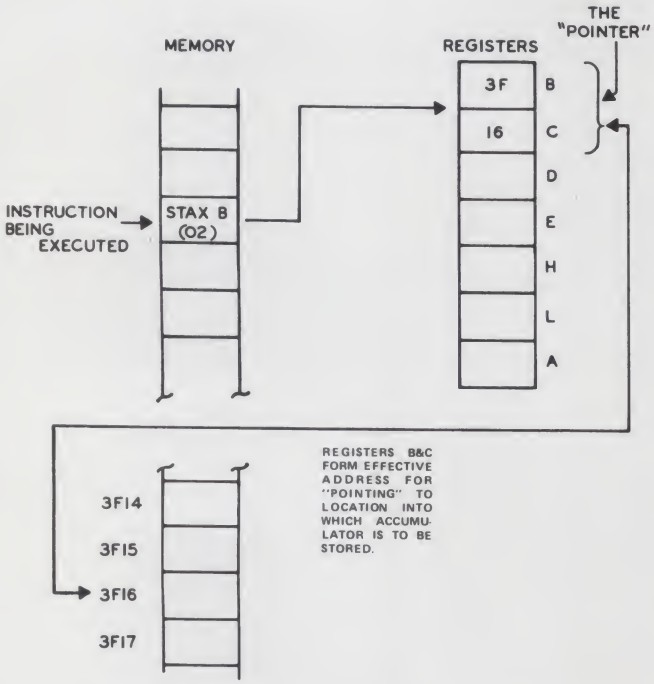
OPERATOR/OPERAND: A statement will consist of an operator — the instruction and an operand — the data to be "operated upon." In the following example, the statement consists of a LoaD a Register instruction (LDA) along with an operand address, which is 500. LDA is *The contents of location 500 is the OPERAND.*

OR GATE: A circuit element whose output is logical if any of the inputs are logical. The truth table and symbol for a 2 input OR gate is shown below.



Inputs		Output
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

POINTER: In its most common useage, a pointer refers to an address which is used to identify the location in memory where something can be stored or found. The object "pointed to" could be an ASCII character, a table of some kind, a variable, an interrupt handling routine, or just about any kind of program or data structure imaginable.



An example of an operation using a pointer would be the 8080 instruction, STAX B. This instruction is used for storing the contents of the A Register (accumulator) into a memory location specified by the B and C Registers. In this case, the B and C Registers are being used as *pointers* (i.e., pointing to the location in memory in which A Register should be stored).

Note that in some computer literature the word *vector* is used interchangeably with the word *pointer* and has the same general connotations as described above.

PHOTODIODE: A special type of diode which allows a current flow proportional to the amount of light striking it. This may be thought of as a transistor with light providing the base current. Almost all transistors will, when removed from their protective cases, function as photodiodes.

PRIORITIZE: In computer systems, to prioritize means to arrange a group of similar things in some order based upon their relative importance to one another. The ordering can be in terms of either location or time.

PROM: Programmable Read Only Memory. Programming is accomplished by burning out fusable links at each memory cell. This is done by flowing a large current through the nichrome link. The programming is permanent and is a useful way to store programs, but manual programming can be very tedious and mistakes cannot be erased. Common types are the 8223 and 825123 (32 8 bit words).

RAM: Random Access Memory. Also called Read/Write memory. These units consist of memory cells which can have information (either a 1 or a 0) stored in them. The individual cells may be accessed in any order (hence the term random. This, compared to shift register memory where all of the locations must be examined sequentially. There are two types of RAM: Static RAM and Dynamic RAM. Each memory cell in a static RAM is composed of a flip-flop storage element. As long as the power is not interrupted the memory cell will retain the information. In a dynamic RAM the storage cell is a capacitor which must be continually recharged to maintain a one state. This recharge process is called refreshing and is done by cycling through the address lines. Dynamic RAMs take less circuitry and therefore are more dense than static RAMs, although the refresh does require some external circuitry. Presently, dynamic RAMs of up to 16,384 bits/chip and static RAMs of up to 4096 bits/chip are available.

ROM: Abbreviation for Read Only Memory. A memory device which has a fixed content in each location. A simple example is a diode encoding matrix for a keyboard. The most common devices are integrated circuits which have the memory contents set by the mask used in manufacture. Integrated circuit ROMs are available with up to 48,000 bits per chip.
Also see RAM, EROM, PROM, EAROM.

SOURCE PROGRAM: A program written in *symbolic code* for an Assembler, Interpreter, or Compiler. For example, with a Motorola 6800-based system an *assembly-language* program could be written using symbolic code, such as LDS to specify loading the stack pointer. This mnemonic (LDS), along with all the other symbolic code making up the program, will then be translated, using an *assembler program*, into *machine language* (or *object code*). In the case of the LDS instruction, the equivalent machine code would be hexadecimal BE.

In the case of a BASIC interpreter or compiler the source code would be the BASIC statements and commands making up a program. Naturally, these will have to be translated into machine language (object code) by the interpreter or compiler before execution can take place.

The SOURCE PROGRAM (Written in symbolic code)

Location	Machine Code	Symbolic Address	Instruction Mnemonic	Operand Data/Addr	Comment Field
1000	C6 08	BCD	LDA B	*NB	
1002	FE 01 00		LDX	ADDR	LOAD DATA ADDR
1005	0C		CLC		
1006	A6 07	NEXT	LDA A	NB-1,X	

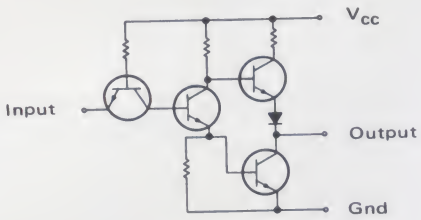
The Executable Machine Code (or, OBJECT CODE)

SWTPC: Abbreviation for Southwest Technical Products Corporation, a company specializing in kits for the hobbyist. They introduced the TVT-1 (a Television typewriter which displays characters from a keyboard on a standard television screen) in 1973, an improved version, the TVT-II in 1974, the 6800 computer using a Motorola 6800 MPU in 1975 and two versions of BASIC for the 6800 in 1976.

SYNCHRONOUS OPERATION of a system is governed by (and synchronized to) a master timing source. Synchronous means constant frequency and phase. For example, in a computer the data transfers occur at definite times relative to the system clock.
Also see Asynchronous operation.

TTL: Abbreviation for Transistor Transistor Logic. This refers to a family of integrated circuit logic elements with a specific output structure. The output stage consists of a pair of

transistor as shown in the diagram.



TTL Inverter

In a zero state (output low) the upper transistor is off and the lower transistor is on. In the one state (output high) the lower transistor is off and the upper transistor is on. In this way the output stage has the ability to source or sink a large amount of current if necessary, but does not have to dissipate this power if not needed since only one transistor is on at a time. In addition the active current source of TTL allows higher speeds.
Also see CMOS, Bipolar.

TTY: An abbreviation for Teletype* (or other teletypewriters), a machine which looks much like an electric typewriter and is used for communicating with a computer. These units are often equipped with a paper tape reader and punch. The advantages of the unit are the hard copy output and the ability to read and punch paper tape. Their disadvantages are slow speed (10 characters per second), high cost (about $1500 new), high noise level and need for frequent maintenance.
*Registered trademark of the Teletype Corporation.

UART: Acronymn for Universal Asynchronous Receiver – Transmitter. As the name implies, the unit has a transmitter and an independent receiver. The UART transmitter accepts parallel input data and transmits it serially. The receiver converts serial input data to parallel output data. The transmitter inserts the desired number of start bits at the beginning of the transmitted word and stop bits at the end of the word. If desired, a parity bit is inserted. The data is then transmitted out at a data rate (baud rate) determined by the transmitter data clock. The receiver section looks for a valid start bit at the beginning of a serial data word. When one is received, the receiver converts the incoming data to a parallel output word. The receiver only responds to data input at a rate determined by the receiver data clock. In addition, the receiver checks for parity and data overrun errors (new data received before the last data was read).

The primary use for UARTs is for interfacing serial data devices to parallel devices. Examples are TTY to computer, computer to telephone line, and computer to tape recorder.

V_{cc}: The collector power supply voltage. In TTL this is +5 volts. Other terms used for the various supplies are V_{BB}, V_{DD} and V_{SS} (B, D, and S standing for bias, drain and source respectively). Typical values for these are V_{BB} = -5, V_{DD} = +12 and V_{SS} = 0. The current flowing from these power supplies is denoted by a capital I with the subscript appropriate to the particular supply (I_{CC}, I_{DD}, I_{BB}).

VECTOR: – See "Pointer"

WORKSPACE is a loosely defined term usually taken to mean the amount of memory required by a program, over and above the amount of memory required to store the program itself. Workspace is typically used for input/output device buffer areas and for various other locations required by a program during its execution.

MORSE CODE TAPES?

Morse code is about as handy for a computerist as a Babbage computer . . . you'll probably want to learn this lovely old method of communications.

Since you'll undoubtedly be wanting to get in touch with some other computerists, and you may be too chintzy to pay Ma Bell her due day in and day out, the chances are good that you'll eventually want to get a Ham license. You have to know the code for this . . . a hangover from the 20's and 30's, which is still being kept alive within the hallowed halls of the FCC.

If you want to learn the code then the 73 code tapes are by far the easiest way known to man to do it. The #5 cassette (one hour) will get you familiar with all of the letters, numbers and punctuation you'll need to convince the FCC to let you Ham. The #6 cassette is strictly practice and is about as difficult as six word per minute Morse code can get. You'll love it.

#5 Intro to Morse Code	$4.95
#6 Back Breaker Practice	$4.95
#13 13 WPM Code Practice	$4.95

We do have some other code tapes available . . . 10 WPM for Canadian licenses, #20 WPM for the Amateur Extra license, and 25 WPM for masochists. $4.95 each.

THEORY TAPES

A set of four one hour tapes has been developed for class use to teach the fundamentals of electricity and electronics, enough to get you through the Novice license. The whole set is $15.95 . . . a bargain. These are great for use in the car, during lunch breaks, etc., if you have a small cassette player.

73 CODE•COURSE 5 WPM 73 CODE•COURSE 13 WPM

MORSE CODE TAPES @ $4.95

☐ #5 ☐ #6 ☐ #13 ☐ #10 ☐ #20 ☐ #25

☐ THEORY TAPES SET OF 4 – $15.95

Name _____

Address _____

City _____ State _____ Zip _____

_____ Enclosed ☐ Cash ☐ Check ☐ Money Order

☐ American Express ☐ Master Charge ☐ BankAmericard

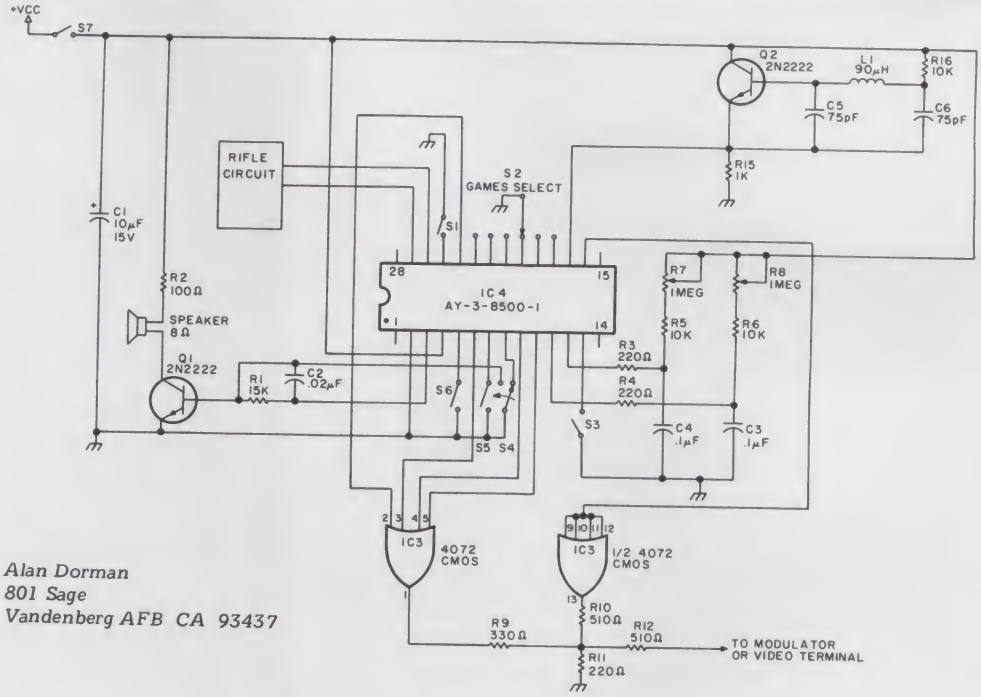
Card # _____

Interbank # _____ Exp. date _____

Signature _____ 1/77

kilobaud PETERBOROUGH NH 03458

Fig. 1. Main schematic with the optional 2 MHz clock shown. Four "C" cells work well for Vcc.



Soon after Al finished building this video game he brought it over to my house and left it for two days. I had to practically pry it out of the hands of my kids when the time came for him to take it. Go ahead and build one for the family for Christmas . . . they'll love ya for it. — John.

Alan Dorman
801 Sage
Vandenberg AFB CA 93437

Six Games on a Chip

... TV tournament time

It's finally here . . . a one chip "pong." And it's not just a single-game-wham-bam-get-bored-quick type of chip. This one has six separate and unique games. So when you think you have

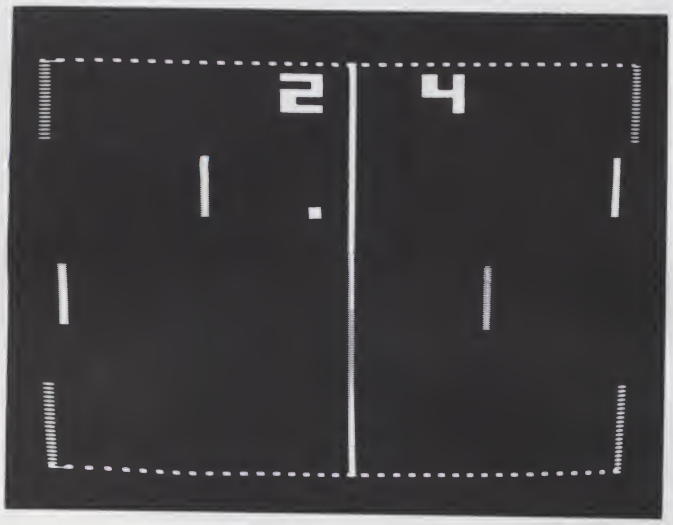


Photo 1. Hockey game.

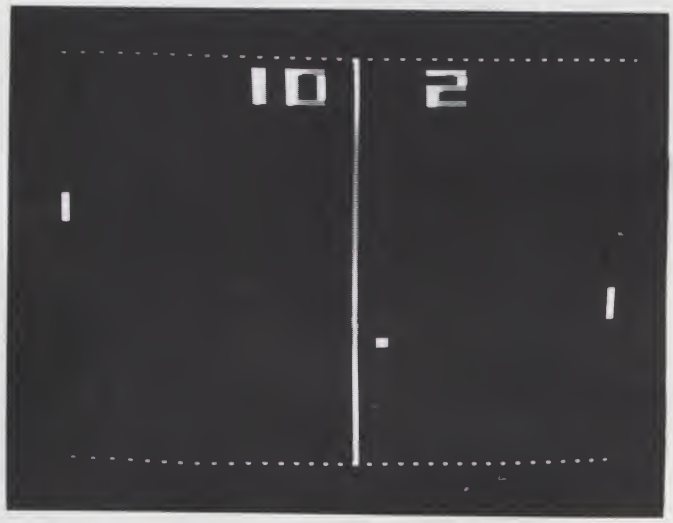


Photo 2. Tennis game with small paddles.

SOCKETS

STANDARD, SOLDER, TIN

	1-24	25-49	50-100
14 Pin	0.27	0.25	0.24
16 Pin	0.30	0.27	0.25
18 Pin	0.35	0.32	0.30
24 Pin	0.49	0.45	0.42
28 Pin	0.99	0.90	0.81
36 Pin	1.39	1.26	1.15
40 Pin	1.59	1.45	1.30

STANDARD, SOLDER, GOLD

	1-24	25-49	50-100
8 Pin	0.30	0.27	0.24
14 Pin	0.35	0.32	0.29
16 Pin	0.38	0.35	0.32
18 Pin	0.52	0.47	0.43
24 Pin	0.70	0.63	0.57
28 Pin	1.10	1.00	0.90
36 Pin	1.75	1.40	1.26
40 Pin	1.75	1.59	1.45

WIREWRAP, GOLD (Level No.3)

	1-24	25-49	50-100
10 Pin	0.45	0.41	0.37
14 Pin	0.39	0.38	0.37
16 Pin	0.43	0.42	0.41
18 Pin	0.75	0.68	0.62
24 Pin	1.05	0.95	0.85
28 Pin	1.40	1.25	1.10
36 Pin	1.59	1.45	1.30
40 Pin	1.75	1.55	1.40

LOW PROFILE, SOLDER, TIN

8 Pin	0.16	0.15	0.14
14 Pin	0.19	0.18	0.17
16 Pin	0.21	0.20	0.19
18 Pin	0.28	0.27	0.26
22 Pin	0.36	0.35	0.34
24 Pin	0.37	0.36	0.35
28 Pin	0.44	0.43	0.42
36 Pin	0.59	0.58	0.57
40 Pin	0.62	0.61	0.60

SAMS BOOKS

TTL COOKBOOK
No.21035 $8.95

IC OP-AMP COOKBOOK
No.20969 $12.95

SECOND CLASS RADIOTELEPHONE LICENSE HANDBOOK, 5TH Ed.
No.21111 $7.50

TRANSISTOR SUBSTITUTION HANDBOOK, 15th Ed.
No.21169 $4.50

REFERENCE DATA FOR RADIO ENGINEERS, 6th Ed.
No.21218 $30.00

BASIC ELECTRICITY/ELECTRONICS SERIES, Vol.1:BASIC PRINCIPALS AND APPLICATIONS
No.20167 $5.50

TV TYPEWRITER COOKBOOK
No.21313 $9.95

TRANSISTOR SPECIFICATIONS MANUAL, 7th Ed.
No.21208 $5.95

FIRST-CLASS RADIOTELEPHONE LICENSE HANDBOOK, 4th Ed.
No.21144 $7.50

SEMICONDUCTOR REPLACEMENT GUIDE
No.21092 $3.95

BUILDING & INSTALLING ELECTRONIC INTRUSION ALARMS
No.20929 $4.50

UNDERSTANDING IC OPERATIONAL AMPLIFIERS
No.20855 $3.95

HOW TO USE IC LOGIC ELEMENTS, 2nd Ed.
No.21081 $4.50

UNDERSTANDING CMOS INTERGRATED CIRCUITS
No.21129 $4.95

SOLID-STATE Digital Auto Clock Build-It-Yourself Kit

FROM



COMPLETE KIT
— ONLY $40.00

Our Quality Is Best!

NEW 74C00 SERIES CMOS

74C00N	.38	74C74N	.90	74C157N	3.40	74C195N	1.80
74C002N	.40	74C76N	1.00	74C160N	2.10	74C221N	2.10
74C004N	.60	74C83N	2.75	74C161N	2.10	74C901N	1.00
74C008N	.40	74C85N	2.75	74C162N	2.10	74C902N	1.00
74C10N	.50	74C86N	1.00	74C163N	2.10	74C903N	1.00
74C14N	2.20	74C89N	14.00	74C164N	1.80	74C904N	1.00
74C20N	.50	74C90N	2.70	74C165N	1.80	74C906N	1.00
74C30N	.50	74C93N	2.60	74C173N	1.80	74C907N	1.00
74C32N	.50	74C95N	1.80	74C174N	1.70	74C914N	2.70
74C42N	2.10	74C107N	1.90	74C175N	1.70	80C95	1.00
74C48N	3.30	74C151N	3.80	74C192N	2.10	80C97	1.00
74C73N	1.20	74C154N	5.70	74C193N	2.10		

DISPLAYS — (COMMON ANODE) Data Sheet EP1000 $.25

LARGE 1"

EP1000	RED	5.50
3 pcs	EP1000	15.00
5 pcs	EP1000	22.50

LITRONIX .3"

DL707	RED	2.35

XCITON .3"

XAN72	RED	2.00
XAN52	GREEN	2.00
XAN82	YELLOW	2.00

LITRONIX .6"

DL747	RED	2.50

.125" dia.

209	RED	$.25
209	YELLOW	.35
209	GREEN	.35

LEDs .160" dia.

216	RED	$.25
216	YELLOW	.30
216	GREEN	.30

LOW PROFILE .200" dia.

226	RED	$.25
226	YELLOW	.30
226	GREEN	.30
226	ORANGE	.30

.200" dia.

5053	RED	$.35
5053	YELLOW	.40
5053	GREEN	.40
5053	ORANGE	.40

.200" dia.

220	RED	$.25
220	YELLOW	.30
220	GREEN	.30

.12" dia.

MV50	RED	$.30

OPTO COUPLER
MCT-2 1.50

CONTINENTAL SPECIALTIES



PROTO BOARDS & EXPERMENTERS

Build & test circuits as fast as you think!

PB-6	$15.95	PB103	$59.95
PB100	19.95	PB104	79.95
PB101	29.95	EX300	9.95
PB102	39.95	EX600	10.95



PROTO BOARDS

With built-in regulated short-proof power supplies

PB203	5V, 1 amp regulated power supply	$ 75.00
PB203A	5V, 1 amp and ±15V, ½ amp regulated power supply	120.00

LOGIC MONITOR 1

Simultaneously displays static and dynamic logic states of DTL, TTL, HTL or CMOS DIP ICs. Pocket size. $74.95

LOGIC MONITOR 2

Displays static and dynamic states of digital ICs such as C-MOS, HTL, TTL, DTL & RTL. Selectable threshold control. $124.95

LOGIC PROBE LP-1

Compact, circuit-powered multifunction logic probe. Multi-family compatibility. DTL/TTL/HTL/CMOS. Traces logic levels and pulses through digital circuitry. $44.95.

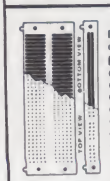
PROTO-CLIP

For power-on/hands off signal tracing. Bring IC leads up from PC board surface for fast trouble-shooting.

PC14	14-pin	$4.50
PC16	16-pin	4.75
PC24	24-pin	8.50
PC40	40-pin	13.75

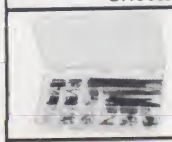


SOCKETS & BUS STRIPS



Plug-in, wire, test, modify or expand without patch cords or solder. Snap together to form breadboard needed.

PN/Description	Term'ls	Price
QT59S Socket	118	$12.50
QT59B Bus	20	2.50
QT47S Socket	94	10.00
QT47B Bus	16	2.25
QT35S Socket	70	8.50
QT35B Bus	12	2.00
QT18S Socket	36	4.75
QT12S Socket	24	3.75
QT8S Socket	16	3.25
QT7S Socket	14	3.00



BREADBOARD JUMPER WIRE KIT

Each kit contains 350 wires cut to 14 different lengths from 0.1" to 5.0". Each wire is stripped and leads are bent 90° for easy insertion. Wire length is classified by color coding. All wire is solid tinned 22 gauge with PVC insulation. The wires come packed in a convenient plastic box.
BK-1 $9.95

CARBON FILM RESISTORS (5%)

Only in Multiples of 100 pcs per value (ohms)
¼W . . . $1.69 per 100
½W . . . $1.79 per 100

10	100	1.0K	10K	100K	1.0M
11	110	1.1K	11K	110K	1.1M
12	120	1.2K	12K	120K	1.2M
13	130	1.3K	13K	130K	1.3M
15	150	1.5K	15K	150K	1.5M
16	160	1.6K	16K	160K	1.6M
18	180	1.8K	18K	180K	1.8M
20	200	2.0K	20K	200K	2.0M
22	220	2.2K	22K	220K	2.2M
24	240	2.4K	24K	240K	2.4M
27	270	2.7K	27K	270K	2.7M
30	300	3.0K	30K	300K	3.0M
33	330	3.3K	33K	330K	3.3M
36	360	3.6K	36K	360K	3.6M
39	390	3.9K	39K	390K	3.9M
43	430	4.3K	43K	430K	4.3M
47	470	4.7K	47K	470K	4.7M
51	510	5.1K	51K	510K	5.1M
56	560	5.6K	56K	560K	5.6M
62	620	6.2K	62K	620K	6.2M
68	680	6.8K	68K	680K	6.8M
75	750	7.5K	75K	750K	7.5M
82	820	8.2K	82K	820K	8.2M
91	910	9.1K	91K	910K	9.1M

ALUMINUM ELECTROLYTIC CAPACITORS



(RADIAL LEAD)

MFD	10V	16V	25V	35V	50V
1	.12	.13	.14	.15	.16
5	.13	.14	.17	.20	.24
10	.14	.15	.18	.22	.26
22	.15	.16	.21	.23	.28
33	.16	.17	.22	.24	.30
47	.17	.18	.24	.25	.32
100	.18	.20	.26	.30	.35
220	.23	.25	.35	.40	.50
330	.30	.33	.40	.45	.55
470	.35	.40	.45	.60	.80
1000	—	—	.65	.80	1.20
2200	—	—	1.30	1.60	2.50

RESISTOR ASSORTMENTS

Each assortment contains 5 pcs each of 10 different values.
Values included are shown in (Ohms)
¼W, 5% (50 pcs total) . . . $1.75
½W, 5% (50 pcs total) . . . $1.85

Asst. 1: 10, 12, 15, 18, 22, 27, 33, 39, 47, 56 ohm
Asst. 2: 68, 82, 100, 120, 150, 180, 220, 270, 330, 390 ohm
Asst. 3: 470, 560, 680, 820 ohm, 1K, 1.2K, 1.5K, 1.8K, 2.2K, 2.7K
Asst. 4: 3.3K, 3.9K, 4.7K, 5.6K, 6.8K, 8.2K, 10K, 12K, 15K, 18K
Asst. 5: 22K, 27K, 33K, 39K, 47K, 56K, 68K, 82K, 100K, 120K
Asst. 6: 150K, 180K, 220K, 270K, 330K, 390K, 470K, 560K, 680K, 820K
Asst. 7: 1M, 1.2M, 1.5M, 1.8M, 2.2M, 2.7M, 3.3M, 3.9M, 4.7M, 5.6M
Asst. 8R: (All seven assortments above)
¼W, 5%, 350 pcs total: $10.95—½W, 5%, 350 pcs total: $11.55

JANUARY SPECIALS

MICROPROCESSOR CIRCUITS

8080A	23.90	1402A	10.00	P2102-1	2.50
AM9080A	34.95	1403A	3.00	P2102A-2	5.20
AM2901DC	31.00	1404A	3.00	P2111	3.80
C1702A	11.00	1405A	3.40	P2111-1	4.80
AM2902PC	3.80	2809PC	4.00	P2112-2	4.80
AM2905PC	10.10	AM9101ADC	8.00	P2405	8.00
AM2906PC	11.20	AM9101APC	6.00	P3101	4.00
AM2907PC	9.75	AM9130ADC	28.40	P3101A	4.10
AM2909PC	25.96	AM9140ADC	28.40	P8212	3.60
AM2911DC	19.96	P8101	4.85	AM2502PC	5.80
AM2918PC	5.40	P8102	4.17	AM2504DC	7.50
AM29720PC	6.25	P8111	4.85	AM2505PC	12.00
AM29721PC	6.25	P1101A	2.20	AM2602PC	1.90
AM685DC	10.00	P2101	3.60	AM2804PC	4.00

Valid only until January 31, 1977

7400N TTL

7400	.18	7432	.32	7480	.80	74128	.60
7401	.20	7433	.44	7482	1.50	74132	1.50
7402	.18	7437	.36	7483	.82	74136	1.80
7403	.20	7438	.35	7485	1.20	74141	1.70
7404	.21	7439	.36	7486	.44	74145	1.20
7405	.23	7440	.20	7489	2.20	74147	2.00
7406	.30	7441	.90	7490	.56	74148	1.80
7407	.36	7442	.66	7492	.60	74150	1.70
7408	.28	7443	.78	7493	.62	74151	1.20
7409	.28	7444	1.05	7494	.88	74152	1.20
7410	.22	7445	1.05	7495	.90	74153	1.20
7411	.26	7446	1.05	7496	.80	74154	1.20
7412	.36	7447	.98	7497	5.00	74155	1.10
7413	.54	7448	.98	74100	1.20	74156	1.20
7414	.98	7450	.15	74105	.90	74157	1.20
7416	.42	7451	.16	74107	.40	74158	1.40
7417	.42	7453	.18	74109	.45	74160	1.30
7420	.15	7454	.18	74110	.80	74161	1.40
7421	.38	7459	.25	74111	1.00	74162	1.90
7422	.60	7460	.19	74116	2.00	74163	1.50
7423	.35	7470	.40	74118	2.00	74164	1.50
7425	.40	7472	.40	74121	.60	74165	1.60
7426	.30	7473	.42	74122	.48	74166	1.60
7427	.36	7474	.42	74123	.95	74170	2.00
7428	.50	7475	.70	74125	.55	74173	1.70
7430	.26	7476	.44	74126	.60	74174	1.20
						74298	1.80

74LS00

74LS00N	.36	74LS51N	.44	74LS132N	1.44	74LS193N	1.70
74LS01N	.44	74LS54N	.44	74LS136N	.74	74LS194AJ	2.20
74LS02N	.36	74LS55N	.44	74LS138N	1.60	74LS195AJ	2.20
74LS03N	.50	74LS73N	.58	74LS139N	1.95	74LS196N	2.76
74LS04N	.44	74LS74N	.58	74LS145N	1.64	74LS197N	2.84
74LS05N	.44	74LS75N	.79	74LS151N	1.55	74LS221N	1.76
74LS08N	.36	74LS76N	.62	74LS153N	1.30	74LS241N	3.00
74LS09N	.44	74LS78N	.92	74LS157N	1.30	74LS251N	2.06
74LS10N	.36	74LS83AN	1.44	74LS158N	1.68	74LS253N	2.40
74LS11N	.44	74LS85AN	2.10	74LS160N	3.00	74LS257N	1.89
74LS15N	.44	74LS86AN	.56	74LS161N	1.50	74LS258N	2.00
74LS20N	.44	74LS90AN	1.14	74LS162N	1.60	74LS260N	.44
74LS21N	.44	74LS92AN	.86	74LS163N	1.50	74LS266N	.72
74LS22N	.44	74LS93AN	.86	74LS164N	1.60	74LS283N	2.00
74LS27N	.44	74LS95AN	2.19	74LS170N	3.20	74LS290N	1.76
74LS30N	.36	74LS96AN	1.24	74LS174N	1.30	74LS293N	1.76
74LS32N	.44	74LS107N	.56	74LS175N	1.30	74LS299N	7.20
74LS33N	.56	74LS109N	.64	74LS181N	5.90	74LS375N	.82
74LS37N	.50	74LS112N	.64	74LS190N	2.85	74LS378N	1.60
74LS38N	.58	74LS113N	.64	74LS191N	2.70	74LS386N	.72
74LS40N	.44	74LS114N	.92	74LS192N	1.90	74LS670N	3.95
74LS42N	.92	74LS123N	1.10				

HIGH SPEED TTL

74H00N	.33	74H08N	.40	74H73N	.80	74H106N	.95
74H04N	.33	74H10N	.33	74H74N	.80		

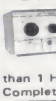
SCHOTTKY TTL

74S00	.44	74S11	.65	74S64	.60	74S174	1.00
74S02	.60	74S20	.65	74S74	.90	74S133	.80
74S04	.55	74S30	.80	74S76	1.15	74S138	2.20
74S08	.80	74S32	.80	74S112	1.00	74S151	2.20
74S10	.55	74S40	.65	74S113	1.50	74S160	3.90
						74S257	2.40

CMOS

4000AE	.20	4023AE	.24	4051AE	1.70	4093AE	1.65
4001AE	.24	4024AE	.95	4052AE	1.40	4095BE	1.95
4002AE	.24	4025AE	.24	4053AE	1.70	4098AE	2.50
4006AE	1.20	4026AE	2.50	4055AE	1.95	4099AE	2.90
4007AE	.24	4027AE	.55	4056AE	1.99	4502AE	1.50
4008AE	1.40	4028AE	1.00	4060AE	2.10	4507BE	.60
4009AE	.50	4029AE	1.20	4063AE	2.50	4508BE	2.20
4010AE	.50	4030AE	.48	4066AE	.44	4510AE	1.40
4011AE	.24	4033AE	2.00	4068AE	.44	4511BE	1.30
4012AE	.24	4035AE	1.30	4069AE	.44	4512BE	1.30
4013AE	.40	4040AE	1.20	4070BE	.60	4514AE	5.00
4014AE	1.20	4041AE	1.25	4071AE	.45	4515AE	5.00
4015AE	1.30	4042AE	1.20	4072AE	.34	4516AE	1.75
4016AE	.80	4043AE	.95	4073AE	.40	4518AE	1.50
4017AE	1.05	4044AE	1.10	4075AE	.40	4520AE	1.28
4018AE	1.24	4046AE	3.10	4076AE	1.24	4528AE	2.20
4019AE	.54	4047AE	2.50	4077AE	.40	4585AE	2.05
4020AE	1.45	4048AE	1.43	4078AE	.40	4901AE	.92
4021AE	1.30	4049AE	.58	4081BE	.40	4911AE	.32
4022AE	1.05	4050AE	.58	4082AE	.39		

MATSUO DIPPED TANTALUM CAPACITORS



MF	V	$	MF	V	$
.1	35	.33	6.8	16	.40
.15	35	.33	6.8	35	.45
.22	35	.33	10.0	16	.42
.33	35	.33	10.0	25	.45
.47	35	.33	10.0	35	.90
.68	35	.33	15.0	20	.45
1.0	35	.33	15.0	35	1.32
1.5	35	.40	22.0	16	.45
2.2	20	.33	33.0	20	1.32
2.2	35	.40	47.0	20	1.53
3.3	35	.42	68.0	16	1.62
4.7	35	.45			

PLESSEY POLYESTER (CAPACITOR) MINI BOX



MF	V	$	MF	V	$
.001	1000	.14	.039	250	.15
.0012	1000	.14	.047	250	.15
.0015	1000	.14	.056	250	.15
.0018	1000	.14	.068	250	.15
.0022	1000	.14	.082	250	.17
.0027	1000	.14	.1	100	.17
.0033	1000	.14	.12	100	.18
.0039	630	.14	.15	100	.20
.0047	630	.14	.18	100	.21
.0056	630	.14	.22	100	.23
.0068	630	.14	.27	100	.26
.0082	630	.14	.33	100	.30
.01	630	.14	.39	100	.33
.012	630	.14	.47	100	.36
.015	400	.15	.56	100	.44
.018	400	.15	.68	100	.47
.022	400	.15	.82	100	.54
.027	400	.15	1.0	100	.60
.033	250	.15			

POWER SUPPLIES—ADTECH

Model No.	Output Vdc	Output Amps	
APS5-3	5	3.0	
APS12-1.6	12	1.6	
APS15-1.5	15	1.5	30.95
APS24-1	24	1.0	
APS28-0.8	28	0.8	
APS5-6	5	6	
APS12-4	12	4	
APS15-3	15	3	50.40
APS24-2.2	24	2.2	
APS28-2	28	2	
APS5-10	5	10	
APS12-7	12	7	
APS15-6	15	6	71.20
APS24-5	24	5	
APS28-4	28	4	
DAPS12-.75	±12	75	
DAPS15-60	±15	60	41.50
DAPS12-1.5	±12	1.5	
DAPS15-1.3	±15	1.3	

XR FUNCTION GENERATOR



XR-2206KA SPECIAL $16.95
Includes monolithic function generator IC, PC board, and assembly instruction manual.

XR-2206KB SPECIAL $26.95
Same as XR-2206KA and includes external components for PCboard.

PLESSEY SAMPLER

300 METALIZED CAPACITORS
18 DIFFERENT STD. VALUES $26.00

L-15 PULSE/SWEEP FUNCTION GENERATOR KIT

FEATURES: 1 HZ to 100 KHz in 5 ranges, at constant amplitude • Sine and triangle adjustable to 20vp-p into 600 ohms • Sine distortion is .5% typical • Voltage controlled sweep of sine and triangle waves • TTL square wave has a fully adjustable pulse and drives 600 ohms • Frequencies of lower than 1 Hz and greater than 100 kHz are obtainable • short proof
Complete with power supply and metal case $55.00

ANCRONA

Minimum order $10.00. Add $1.00 to cover postage and handling. California Residents add 6% sales tax. **Mail Order** — Send Check or money order to: **P.O. Box 2208S Culver City, CA 90230.** COD, Master Charge and Bank Americard Welcomed. *(Stores do not accept mail or phone orders.)*

• Phone Order — Call: (213) 641-4064

CALIFORNIA ANCRONA
11080 Jefferson Blvd.
Culver City, CA 90230
(213) 390-3595

CALIFORNIA ANCRONA
1300 E. Edinger Ave.
Santa Ana, CA 92705
(714) 547-8424

ARIZONA ANCRONA
4518 E. Broadway
Tucson, AZ 85711
(602) 881-2348

OREGON ANCRONA
1125 N.E. 82nd Ave.
Portland, OR 97220
(503) 254-5541

GEORGIA ANCRONA
3330 Piedmont Rd. NE
Atlanta, GA 30305
(404) 261-7100

CANADA, B.C. ANCRONA
5656 Fraser St.
Vancouver, BC V5W 2Z4
(423) 324-0707

mastered one game, just switch to the next and try your luck. With two types of rifle shoot plus hockey (Photo 1), tennis (Photo 2, shown with the game switched to smaller paddle sizes), squash (Photo 3), and practice to choose from, I'm sure you and your friends will be entertained for many hours. The AY-3-8500-1 MOS chip from General Instruments is a very versatile chip. By using a standard television (with an rf modulator circuit) or direct video output to a video monitor (or modified television), the games can be generated with minimal external circuitry. The MOS circuit is battery operational and has a low current drain of only 30 mA.



Photo 4. The completed (and compact) unit.

Features

Some of the features of this circuit are automatic scoring with a 0-15 score display on the TV screen plus the audio "bink," "bonk," and "boing" sounds necessary for any professional video game. These are not optional features; they are included in the logic of the chip.

The game can provide either 1.9 inch or .95 inch high bats on a 19 inch diagonal television screen. The bats can also be divided into two or four areas of deflection. In the two-angle mode, the top and bottom of the paddle determine the angle of deflection. If the ball hits the top or bottom of the paddle, it will assume the angle of deflection and continue in play. In the four-angle mode, the paddle is divided into four equal sections. The quarter of the paddle which hits the ball defines the new direction for the ball. The direction does not depend on the previous angle at which the ball hit. With the two-angle mode, the top and bottom pairs of the paddle are summed together and only the two shallower angles are used to provide the new direction for the ball.

The ball can be re-served either automatically or manually. In the automatic mode the ball will re-serve after each score. With manual the game will stop serving the ball after each score is made. Ball speed is selectable from 1.30 seconds to .65 seconds for the time taken by the ball to transverse the television screen.

Pin Designations

(1) No connection.
(2) Ground.
(3) Sound output. A hit is equal to a 32 millisecond pulse of 976 Hz. A boundary reflection will give a 32 millisecond pulse of 488 Hz, while

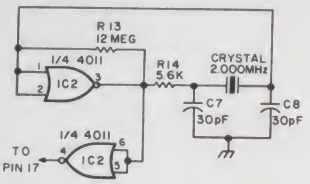


Fig. 2. 2 MHz oscillator that is included in the board layout.

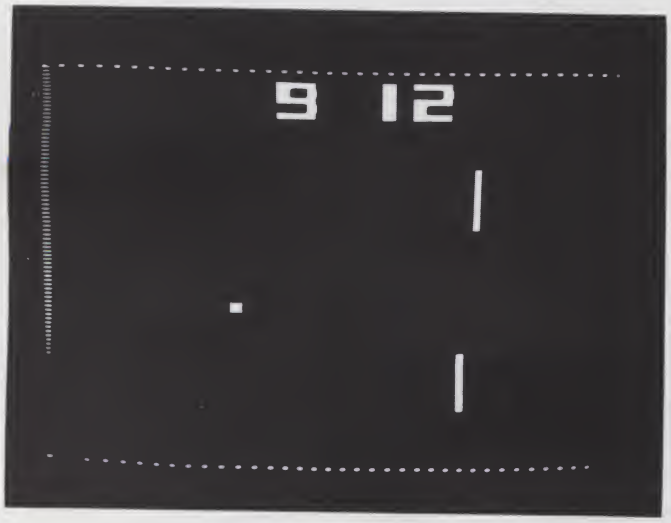


Photo 3. Handball.

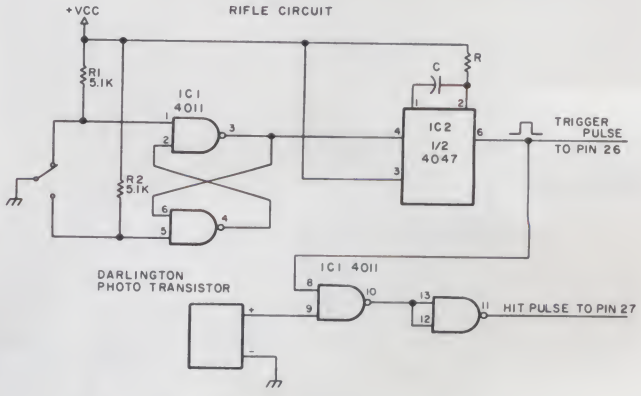


Fig. 3. Schematic for the rifle circuit. I have not done any work on this circuit as of yet, so you are on your own. It looks pretty simple though, and shouldn't give too much trouble.

RONDURE COMPANY

Where We Ship from Inventory the Same Day Your Order Arrives*

AT THE LOWEST PRICE WE'VE EVER OFFERED!

A SELECTRIC TERMINAL COMPLETE WITH RS-232/C INTERFACE AND CERTIFIED FOR MAINTENANCE BY A NATIONAL SERVICE COMPANY. SHIPPED THE SAME DAY WE RECEIVE YOUR CHECK*



$895.00

Specifications

- ☐ Size: 21" wide x 21" deep x 8" high.
- ☐ Power Input: 115 Volt, 60 Hz.
- ☐ Mounting: Tabletop.
- ☐ Interface: RS232C.
- ☐ Weight: 54 lbs.
- ☐ Color: greyish beige; blue.
- ☐ Environment: Normal office conditions.

We Buy and Sell the Following Equipment



MINI-COMPUTERS AND MICRO-COMPUTERS
FORMS HANDLING EQUIPMENT
COMPUTER PERIPHERALS
COMPUTER TERMINALS

Bursters
Tape Cabinets
Disk Pack Cabinets
Keypunch Desk
CRT Tables

PRINTERS, READERS, PUNCHES
TAPE DRIVES, DISK DRIVES

ACCOUSTICAL MODELS ORIGINATE ONLY
USED – UNTESTED
$20.00 ea. 2 for $35

BASE CASES FROM SELECTRIC MECHANISMS SUITABLE FOR MICRO/MINI OR TERMINAL BUILDING BLOCK $7.50

NEW ADDRESS
2522 BUTLER
DALLAS, TEXAS 75235
Phone: (214) 630-4621

TERMS: Check or Money Order. For Modems, Base, Keyboard, Switch Blk., add $2.00 shipping and handling. All others shipping packaging and shipping collect.

*Maintenence limited to cities in which service now offered. Shipped the same day as certified check or money order arrives. When regular checks accompany order, equipment is shipped when regular check clears.

ALSO NOTE: NO EQUIPMENT INCLUDES PRINTS OR DOCUMENTATION (unless stated), NO CONNECTING CORDS OR CONNECTORS. EQUIPMENT IS SHIPPED ON AN AS IS – WHERE IS – BASIS. EXCEPT WHERE EXPRESSLY STATED IN WRITING, NO REPRESENTATION OR WARRANTY IS MADE AS TO THE QUALITY, CONDITION OR WORKING ORDER OF ANY EQUIPMENT OR PART.

a score will give 160 milliseconds of a 1.95 kHz tone.
(4) Vcc, +6 to +7.5 V dc.
(5) 2/4 angles input. When left open, two rebound angles are allowed (±20°). When connected to ground, four rebound angles are selected (±20°, ±40°).
(6) Ball output. The ball video signal is output here.
(7) Ball speed input. When left open, low speed circuit is selected at 1.30 seconds, and when grounded, 0.65 seconds is selected.
(8) Manual serve. When grounded, the play is restarted automatically after each score. When left open, the play stops after each score. The game can be restarted by connecting the input momentarily to ground.
(9,10) Right and left player video outputs.
(11,12) Right bat and left bat inputs. A resistor capacitor network attached to each of these pins controls the position of the respective players.
(13) Bat size input. When open, small bats are selected. When connected to ground, large bats are selected.
(14,15) No connection.
(16) Sync out. Output for horizontal and vertical television sync signals.
(17) Clock input. The 2.00000 MHz master clock is connected to this pin.
(18) Rifle 1.
(19) Rifle 2.
(20) Tennis.
(21) Hockey.
(22) Squash.
(23) Practice.
(24) Score and field output video signals.
(25) Reset input. This pin, when grounded momentarily, will restart the game at 0-0. It is normally left open.
(26) Shot input. Driven by the output of a monostable to indicate a shot has been fired for the rifle circuitry.
(27) Hit input. This input is driven by the output of a monostable (.5 seconds) which is triggered by the shot input when the target is in the sights of the rifle. A positive pulse is needed. *A note on the rifle game:* The score is displayed only after a hit has been registered to prevent the players from shooting at the score to get points.
(28) No connection.

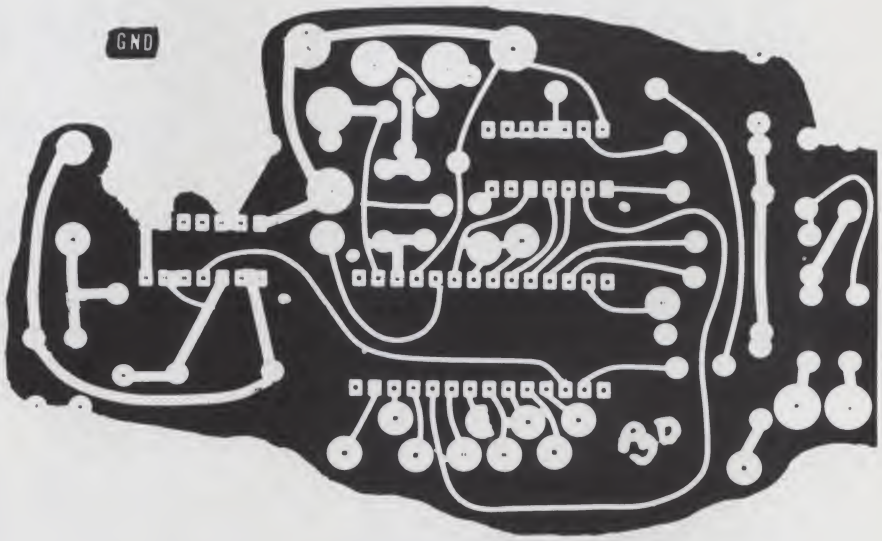


Fig. 4. Board layout.

Circuit Operation

The circuit has two types of clocks that can be used. On the board layout provided (Fig. 4), the crystal oscillator is used. The schematic of the oscillator can be seen in Fig. 2. It is very simple, using two CMOS gates (IC2) for operation. This circuit will give a very stable 2 MHz output for the game chip's timing functions. The optional clock is shown in Fig. 1. Its main advantage is a savings in the cost of the parts needed for construction.

IC3 combines the various video output signals from IC4 and outputs the composite signal through R12. Q1 and associated circuitry provide the audio output. S4 is an SPDT slide switch which grounds the base of Q1 (when in manual serve) so there isn't a steady "boing" when the ball leaves the playing field. R5 through R8 (plus C3 and C4) are used to position the players on the playing field.

The rifle circuit, shown in Fig. 3, connects to pins 26 and 27. I haven't constructed

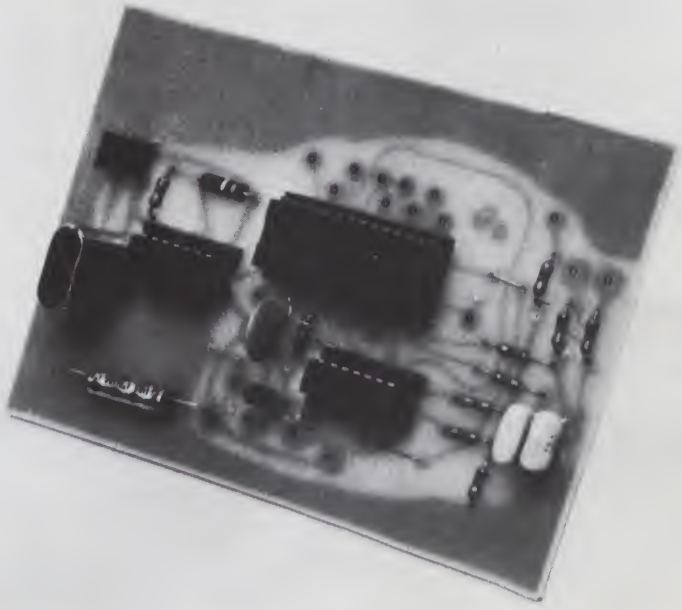


Photo 5. Printed circuit board with components mounted.

"As the designer of the very first frequency counter, I congratulate you on a very neat design at an excellent price. Performance of this counter is really impressive..."

Richard K. Dickey

R. K. Dickey
Professional Electrical Engineer
(Designer of the world's first frequency counter at Berkeley Scientific, 1949)

Thank-you, Mr. Dickey.

Thousands more who own Hufco frequency counters proudly agree.
Since 1974, Hufco has made counters and "counter-offers" that satisfy people to no end. The Hufco combination—a rock-bottom price and sky-high quality—is proving a very popular duo. Here are three good examples why:

A 500 mHz (6-digit) frequency counter for under 40 cents per mHz!

Figure it out. It adds up to unheard-of savings. With guaranteed quality to match and exceed the overpriced brands. Let us prove it to you today.

169.95
(500 mHz kit)
$199.95 assembled



A 30mHz or 250mHz (6-digit) frequency counter — proven best sellers!

69.95
(30mHz kit)
$99.95 assembled

119.95
(250mHz kit)
$139.95 assembled



No wonder people respect the Hufco name.

Incidentally, you'll also find it on economically-priced digital display adapters, voice-operated transmits, power mike adapters, CB/Ham timers, and more.

Interested?
Rush us this coupon today.

Please send me:

- ☐ **500 mHz frequency counter** - 169.95 kit/199.95 assembled (Enclose check or money order.)
- ☐ **250 mHz frequency counter** - 119.95 kit/139.95 assembled (Enclose check or money order.)
- ☐ **30 mHz frequency counter** - 69.95 kit/99.95 assembled (Enclose check or money order.)
- ☐ **Information on other handy *Hufco* products.**

Name _____

Address _____

City/State/Zip _____

Mail to: **Hufco** Box 357, Dept. R, Provo, Utah 84601 801/375-8566

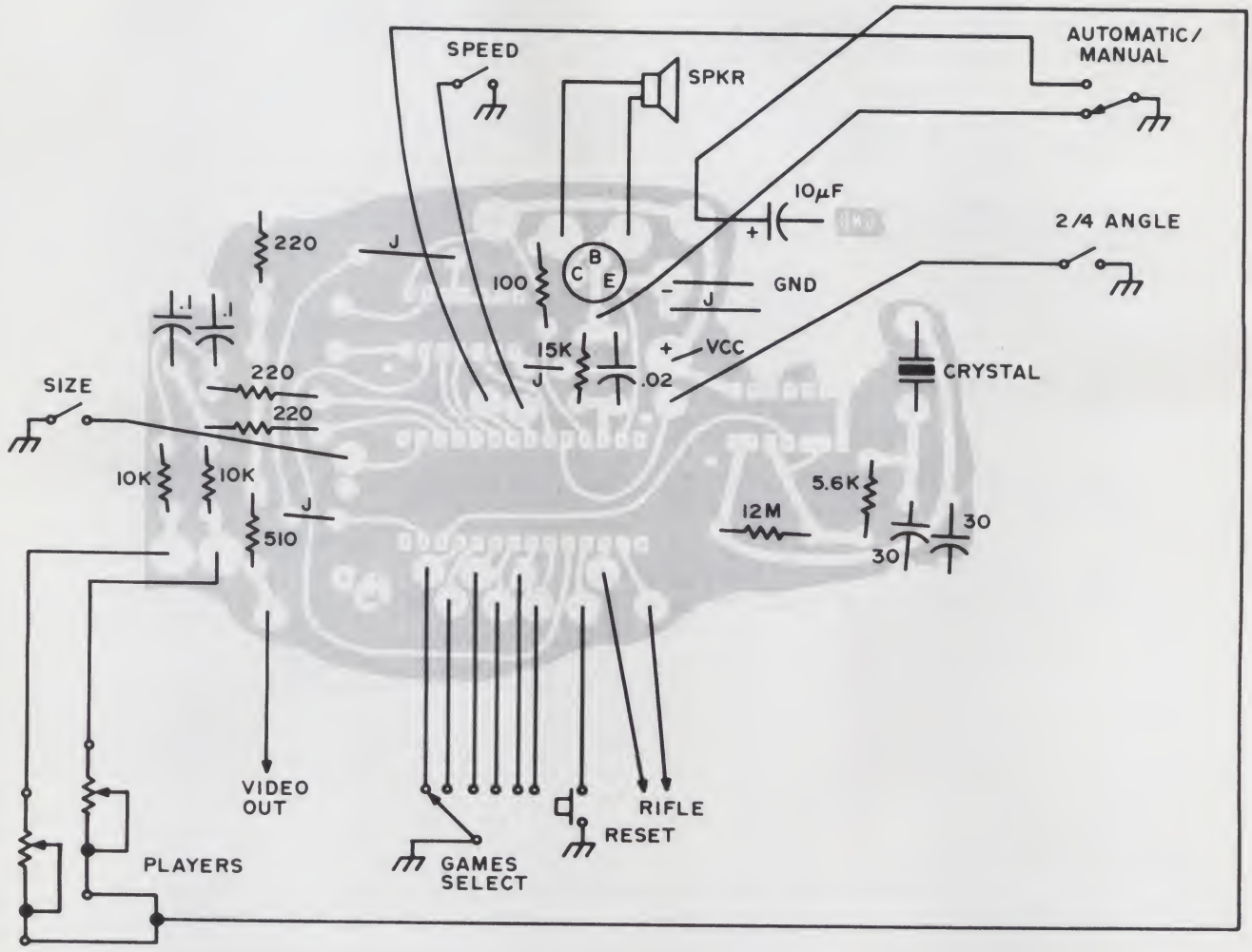


Fig. 5. Component layout.

this portion of the game, but it looks pretty straightforward. A Darlington transistor detects the ball on the screen of the TV. Coincidence of the ball (output of the photo transistor) and a trigger pulse results in a hit pulse being input to pin 27 of IC4. The shot pulse provided by IC1 and IC2 goes to pin 26. The RC network for IC2 should be chosen for an output of .5 seconds.

Construction and Circuit Layout

Any type of construction can be used: PC board, perfboard, or wire wrap. I put my unit in a 7x5 plastic box from the local Radio Shack store (see Photo 4). The PC board layout shown in Fig. 4 will work quite well. While installing parts on the board, good construction practices should be observed to prevent any mechanical or electrical complications. Use Fig. 5 to install the parts and make sure all ICs are oriented in the proper manner. With proper construction, the game should work the first time it is turned on.

One problem was encountered: The audio sounds produced tearing in the picture when used with my version of an rf modulator. This was solved by putting a capacitor in series with the video output to smear the lines slightly so they wouldn't go "squiggly" when a sound was produced. Adjusting the modulator bias helped a lot also. Any of these audio problems can be effectively solved by using the proper type of modulator. I also suggest a three wire system for the controls. With the left and right player controls floating (neither side at ground potential), they will pick up noise and possibly prevent a good visual display. By three wire, I mean two for the pot with an outside shield going to ground. Grounding the metal chassis of your project is also recommended.

Photos 5 and 6 show the completed PC board and the completed unit with PC board installed.

Displaying the Game

When the chip is received, information may be enclosed on building an economical VHF modulator. But be advised that not all modulators meet FCC regulations. Therefore, if you do decide to construct your own, make sure it meets the applicable regulations. Fig. 6 is a sche-

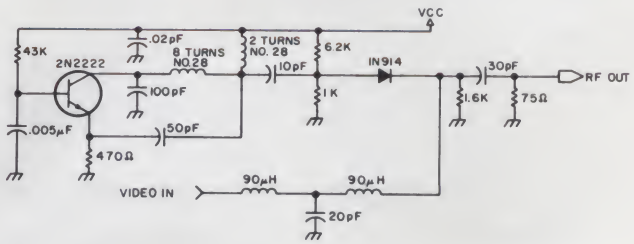


Fig. 6. VHF modulator. NOTE: This modulator circuit is provided for test purposes only *and may not possess the long-term stability or meet FCC requirements for home use.*

SLEP ELECTRONICS IS NOW SHIPPING

ATLAS

210X 80 THRU 10M $679.00
215X160 THRU 15M 679.00
210X OR 215X WITH NOISE BLANKER 719.00
AC CONSOLE 220CS WITH VOX/SEMI BREAK-IN CW 194.00
AC CONSOLE 220CS 147.00
PORTABLE AC SUPPLY 200PS 100.00
DMK PLUG-IN MOBILE MOUNT, WITH DC CABLE 48.00
DCC BATTERY CABLE 12.00
DD-6B DIGITAL DIAL 229.00
MT-1 MOBILE ANTENNA MATCHING TRANSFORMER 27.00
PC-120 NOISE BLANKER CONVERSION KIT 52.00
VX-5 VOX CONVERSION KIT FOR AC CONSOLE 49.00
206 AUXILIARY VFO WITH EXTENDER FREQUENCY COVERAGE FOR ALL ATLAS TRANSCEIVERS 299.00

DRAKE

R-4C RECEIVER 160-10M 599.00
4-NB NOISE BLANKER R-4C 70.00
FILTERS 250HZ, 500HZ, 1.5HZ . ea. 42.00
SPR-4 RECEIVER 629.00
5-NB NOISE BLANKER SPR-4 70.00
DC-PC 12V POWER CORD 5.00
SCC-4 100HZ CALIBRATOR 20.00
CRYSTAL KIT AMATEUR BANDS . . 31.20
SSR-1 GENERAL COVERAGE RECEIVER .5 TO 30MHZ 350.00
DC-PC SSR-1 DC POWER CORD 5.00
T-4XC TRANSMITTER 160-10M ... 599.00
TR-4C TRANSCEIVER 80-10M 599.95
34PNB NOISE BLANKER TR-4C ... 100.00
MMK-3 MOBILE MOUNT 7.00
RV-4C REMOTE VEO TR-4C 120.00
AC-4 POWER SUPPLY FOR ALL DRAKE TRANSMITTER TRANSCEIVERS . . 120.00
DC-4 12V DC MOBILE POWER SUPPLY 135.00
TR-33C 2 METER FM TRANSCEIVER 12 CHANNELS 229.95
AA-10 10 WATT 2 METER AMPLIFIER 49.95
AC-10 POWER SUPPLY TR-22, TR-33, TR-72 49.95
MN-4 ANTENNA MATCH BOX 110.00
MN-2000 ANTENNA MATCHBOX . . 220.00
MS-4 SPEAKER 24.95
W-4 WATTMETER 1.8 TO 54MHZ ... 72.00
7072 HAND-HELD MIKE 19.00
7075 DESK TOP MIKE WITH VOX SWITCH 39.00
RCS-4 REMOTE CONTROL ANTENNA SWITCH 120.00
HS-1 HEADPHONES 10.00
TV-3300LP LOW PASS FILTER 19.95
DSR-2 VLF-HF DIGITAL RECEIVER 2,950.00

TEMPO

TEMPO ONE, 5 BAND SSB TRANSCEIVER 399.00
TEMPO ONE AC POWER SUPPLY ... 99.00
TEMPO VF/ONE EXTERNAL VFO . 109.00
TEMPO DFD/ONE DIGITAL DISPLAY
TEMPO ONE 189.00
TEMPO 2020 SSB/AM TRANSCEIVER 80 TO 10M 115V/12VDC P/S 759.00
TEMPO FMH 2 METER, 6CH HAND-HELD FM TRANSCEIVER 199.00
TEMPO VHF/ONE SYNTHESIZED DIGITAL READ-OUT 2 METER 10 WATT TRANSCEIVER, NO CRYSTALS TO BUY 495.00
TEMPO SSB/ONE SSB ADAPTOR FOR UHF/ONE 225.00
TEMPO 130A10 130 WATTS OUT 10W DRIVE 179.00
TEMPO 80A10 80 WATTS OUT 10W DRIVE 139.00
TEMPO 50A10 50 WATTS OUT 10W DRIVE 99.00

TEN TEC

TRITON IV MODEL 540 TRANSCEIVER 699.00
252G POWER SUPPLY 99.00
252G POWER SUPPLY/VOX 129.00
207 AMMETER 14.00
245 CW FILTER 25.00
ARGONAUT MODEL 509 329.00
405 LINEAR AMPLIFIER 159.00
210 POWER SUPPLY 1 AMP 27.50
251 POWER SUPPLY 9 AMP 79.00
206 CRYSTAL CALIBRATOR 26.95
208 CW FILTER 29.00
HR5A KEYER 38.50
KR20A KEYER 67.50
KR50 KEYER 110.00

BRIMSTONE

MODEL 144 2 METER FM TRANSCEIVER 25 WATT 143 TO 149.99MHZ DIGITAL DIALED 5KHZ STEPS, NO CRYSTALS TO BUY 142MHZ MARS COVERAGE OPTIONAL 650.00

COMCRAFT

CST-50 VHF TWO BAND TRANSCEIVER 2 AND 1¼ METERS, DIGITAL FREQUENCY SYNTHESIS 142 TO 149.995MHZ AND 220 TO 225 MHZ 25 WATTS 869.00

SHURE

526T DESK TOP MICROPHONE WITH PRE-AMP 36.50
444 SSB DESK TOP MICROPHONE WITH OFF-ON VOX SWITCH PTT 34.50
404C HIGH IMPEDANCE MAGNETIC HAND-HELD MICROPHONE FOR SSB PTT, IDEAL FOR SWAN, ATLAS 27.00

MILLEN

92200 ANTENNA MATCHBOX ALL BAND 2KW 199.00
92201 JR ANTENNA MATCHBOX ALL BAND 300 WATTS 138.00
90652 SOLID STATE GRID DIP OSCILLATOR 9 VOLT BATTERY OPERATED WITH COILS 1.7 THRU 300MHZ SUPPLIED 138.00

SWAN

700CX TRANSCEIVER 649.95
117XC CONSOLE SPKR/PS 159.95
14-117 DC MOBILE P/S 189.95
510X CRYSTAL OSCILLATOR 67.95
VX-2 VOX 44.95
FP-1 PHONE PATCH 64.95
SWR-1 POWER/SWR METER 0-1 KW, 3.5 – 150 MHZ SO-239 CONNECTORS 21.95
WM-2000 IN LINE WATTMETER SCALES TO 2KW 49.95
1200X LINEAR AMPLIFIER 1200 WATTS 349.95
MARK II LINEAR AMPLIFIER 2KW 849.95
SS-16B KIT 99.95
45 ALL BAND MANUAL SWITCHING 1KW PEP MOBILE ANTENNA 114.95
742 TRI-BAND 20-40-75 METER ELECTRONICALLY TUNED AUTOMATIC BAND-SWITCHING, 500 W.PEP 79.95

REGENCY TRANSCEIVERS

HR-312 FM 2 METER 30W 269.00
HR-6 FM 6 METER 15W 229.00
HR-220 FM 220MHZ 10W 239.00
HR-440 FM 440 MHZ 10W 349.00
P-110 AC P/S 117V/12VDC REGULATED 5 AMP 49.95

NYE-VIKING

CODE KEYS SPEED-X MODEL 114-310-003 STANDARD KEY, NICKEL PLATED HARDWARE WITH SWITCH 8.25
MODEL 114-310-004 STANDARD KEY, NICKEL PLATE HARDWARE WITH SWITCH AND NAVY KNOB 9.10
SSK-1 DUAL PADDLE SQUEEZE KEY NICKEL PLATED 23.95
MODEL 114-404-002 CODE PRACTICE SET WITH KEY, OSCILLATOR, AMPLIFIER 2" BUILT-IN SPEAKER, HEAVY DUTY BASE, TAKE 9V BATTERY, NOT INCLUDED 18.50

BEARCAT

MODEL 101, 16 CHANNEL SCANNER, 30-50MHZ 146-174MHZ, 416-512MHZ, NO CRYSTALS TO BUY 290.00
W2AU BALUN, 2KW PEP, 3 TO 40 MHZ 1:1 MATCHES 50 OR 75 OHM *UNBALANCED* COAX LINE TO 50 OR 75 OHM BALANCED LOAD, BUILT-IN LIGHTNING ARRESTOR 12.95

Slep Electronics Co.

P.O. BOX 100, HWY. 441, DEPT. 73, OTTO, NORTH CAROLINA 28763

WE PAY SHIPPING VIA U.P.S. OR BEST WAY ON ALL ADVERTISED ITEMS; TRADES TAKEN ON NEW EQUIPMENT. WRITE FOR SPECIAL PACKAGE PRICE ON COMPLETE STATIONS. SATISFACTION GUARANTEED. WE ACCEPT MASTER CHARGE, N.C. RESIDENTS ADD 4% SALES TAX. PHONE BILL SLEP 704-524-7519 MONDAY THRU FRIDAY 8:30 - 6:00 PM.

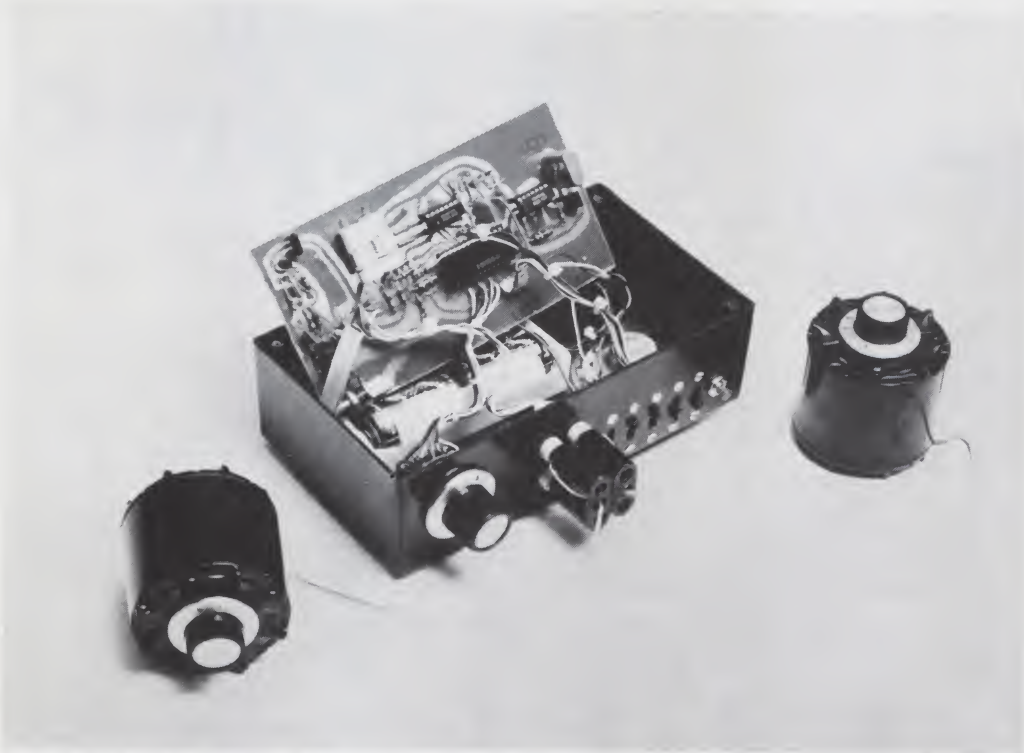


Photo 6. Interior view of the completed unit. (The cases for the paddle-control pots are waterproof ammunition cases.)

matic of a modulator I built for testing the unit. With regard to direct video into a standard television receiver, Don Lancaster's latest book entitled *TV Typewriter Cookbook* is a handy reference on video interfacing. I recommend reading this book if you want a good understanding of television conversion principals, as well as a good understanding of TV typewriters. Many practical applications are included which will be applicable to this game. And, of course, the game has direct video output (with no modification necessary) to any video monitor.

Conclusion

So here it is, a new kind of "pong" chip. It is one of the first of its kind available to the hobbyist. In the near future there will probably be many more. Look how the microprocessor chips started out and how quickly the idea caught on. The AY-3-8500-1 chip may be obtained from MHz Electronics, 2543 N. 32nd St., Phoenix AZ 85008, phone (602) 957-0786, $29.95.■

ALDELCO COMPUTER CENTER NOW OPEN

Kits, Books, Boards, Magazines
Special 2102L1 8 for $17.50. We stock OK Battery Operated Wire Tool $34.95, OK Hand Wire Wrapped Tool $5.95. 7400 ICs CMOS, Timers PLL's. All kinds of transistors, rectifiers, and diodes.

Plus other electronic parts.

ZENERS
1N746 to 1N759 400 Mw ea .25 — 1N4728 to 1N4764 1 w 28

C106B SCR	$.65	CA 3028A Dif. Amp.	$1.50
MPSA14	.90	LM301 Op Amp	.55
2N3055	.99	LM309K Volt Reg.	1.10
MPF102 FET	.45	LM380N Audio Amp	1.75
2N3904 or 2N3906	10/.99	NE540L Power Driver	5.95
2N5496 or 2N6108	.35	NE561B PLL	4.95
MJE340 (2N5655)	1.10	NE562B PLL	4.95
40673 RCA FET	1.55	NE565A PLL	2.50
741 or 709 14 Pin DIP	.25	LM709 Min DIP Op Amp	.45
555 Timer	.75	LM741CE T05 Op Amp	.45
556 Dual 555	1.75	14 or 16 Pin IC Sockets	.30
200 Volt 25 Amp Bridge	1.50	We have 7400 series ICs; send stamp for catalog.	
1N914 - 1N4148	15 for .99		
1N34 - 1N60 - 1N64	10 for .99		

Send for our catalog. Open Mon thru Sat 9 am to 5 pm — Wed till 9 pm.

We quote on any device at any quantity. Min. order $6.00. Out of USA send certified check or money order. Add 5% for shipping.

ALDELCO
2281A Babylon Tnpk Merrick, N.Y. 11566
(516) 378-4555

MODULES FOR ALTAIR AND IMSAI COMPUERS

8K STATIC MEMORY — KIT $295
ASSEMBLED $375
16K STATIC MEMORY — KIT $650
ASSEMBLED $775
WIRE WRAP BOARD — KIT $ 39
EXTENDER BOARD W/C — $ 30
BATTERY BACK-UP BOARD
LESS BATTERIES — KIT $ 55
115V I/O BOARD — KIT $ 149
ASSEMBLED $200
THE 115V I/O BOARD HAS FOUR INPUTS AND FOUR OUTPUTS EACH 115VAC AT 1.5 AMPS. NO LONGER A NEED FOR RELAYS WITH THIS BOARD.
32K DYNAMIC RAM — KIT $ 895
16 K DYNAMIC RAM — KIT $599

To order send check or money order (include $2.50 shipping/handling) to ELECTRONIC ENG. & PRODUCTION SERVICES, Rt.# 2, Louisville, Tennessee. (Tn. users add 6% sales tax) (615)984-9640

□ □ □ □ □ **ENIGMAS-1** □ □ □ □ □
Computer Games in BASIC

Enigmas-1 is a book of computer games taken from my catalog.

The programs in this book are:

GONE FISHING — Go fishing to make money
CONCENTRATION — Match the hidden numbers — two can play
SLOT-MACHINE — It's easy to lose your money
CRAPS DICE GAME WITH DICE PRINT-OUT — Shoot craps and see the dice
TANK ATTACK — Try to defend yourself
STARSHIP (STAR TREK TYPE GAME) — Shoot some Klingons
SHERLOCK HOLMES LOGIC GAME — Chase Professor Moriarty

All my programs in Basic have been written to run in most any Basic compiler-interpreter without any changes.

They will run under Altair 4K or 8K Basic and most any other Basic compiler-interpreter provided you have enough memory to run them.

PRICE: $8.00 ppd. SORRY NO CODs

B. Erickson P.O. Box 11099, Chicago IL 60611

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

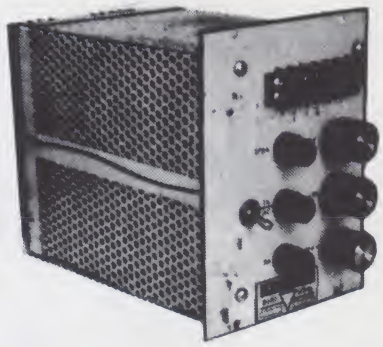


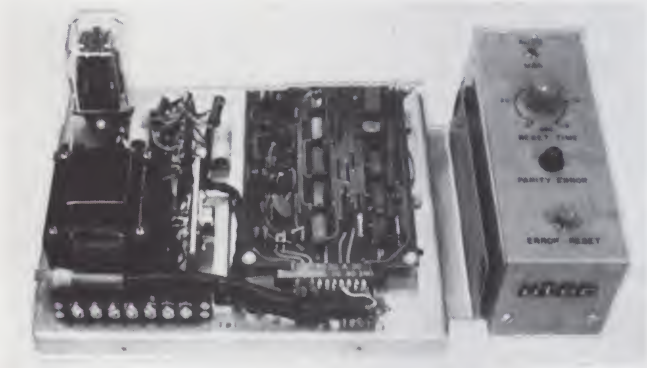
FAX MACHINE BY DATAFAX

We have transmitters and receivers. Used for weather charts, hi speed teletype recording. Normally used over the phone lines. Operational when removed due to upgrading of equipment. Only a few on hand and sold "AS IS." When used over the phone lines from weather data system, will draw full map of the US with cloud cover and also pressure gradients. Due to the weight of these machines they must be shipped via truck as they are around 60 lbs. Made for desk top use. Made by Stewart Warner Elect. Picture is typical unit. When ordering state receiver or transmitter. #FAX ~~$125.00~~ $75.00 FOB Lynn Mass.

GENERAL PURPOSE POWER SUPPLY

A well designed transistorized regulated power supply with many uses. Each voltage adjustable by a pot. Each voltage fused. 115 volts AC 60 cycle input. Output (minus) 12 volts at 1/3 Amp, 12 volts (plus) at 3 Amps, 6 volts at 1 Amp . . . three output voltages. Many uses . . . as battery charger, op amp, 5 volt logic supply, operate your car radio or tape player, CB set, in the house, etc. A commercially built regulated supply for far less the price of a kit. Shipping wgt. 10 lb. #SP-152-L $17.50





PARITY DETECTOR

New packaged, made for RCA, detects even or odd parity, baud rate 110, 150 or 134.46. Built-in logic supply for the ICs, operates from standard 115 V ac. Control panel allows manual or automatic reset mode of operation. Aluminum enclosure (not shown) covers the electronics. TTY compatible. Ship wt. 10 lbs. ~~$16.50~~ $12.50

ZENER 15V 63 Watt $1.00
2.5 Amp 1,000 piv
diode 4/$1.00, 25/$5
Motorola 1N4001 diode 1A,
50 piv 20/$1.00

RT 70/GRC as written up in 73 Magazine, November, 1976. Only $20.00.



SOLAR CELLS

Designed for the space program, these are the highly efficient silicon high output cells. Used for powering equipment, charging batteries. Made by Ion Physics Corp. Each with spec sheet.

Size .394 x .788" 65 mA, .43 V $1.25 12/$12.00
Size .788 x .788" 125 mA, .43 V $1.60 12/$15.00

Meshna

Please add shipping cost on above.

FREE CATALOG SP-9 NOW READY

P.O. Box 62K, E. Lynn, Massachusetts 01904

21L02-1 RAM

500 N. S.

8 for $12.95

82S129-256 x 4 PROM

BI - POLAR--FAST!

$2.50

8080A CPU CHIP

$19.95 By AMD

ZILOG

Z-80 CPU CHIP

$69.95

MM5204 4K EPROM

$7.95

S. D. SALES CO.
P. O. BOX 28810 -
DALLAS, TEXAS 75228

SEE TERMS ON OTHER PAGE!

S.D. SALES CO.

P.O. BOX 28810 -K
DALLAS, TEXAS 75228

Z–80 CPU CARD KIT FOR IMSAI/ALTAIR

$149. KIT

From the same people who brought you the $89.95 4K RAM kit. We were not the first to introduce an IMSAI/ALTAIR compatible Z–80 card, but we do feel that ours has the best design and quality at the lowest price.

The advanced features of the Z–80 such as an expanded set of 158 instructions, 8080A software compatibility, and operation from a single 5VDC supply, are all well known. What makes our card different is the extra care we took in the hardware design. The CPU card will always stop on an M1 state. We also generate TRUE SYNC on card, to insure that the rest of your system functions properly. Dynamic memory refresh and NMI are brought out for your use. Believe it or not, not all of our competitors have gone to the extra trouble of doing this.

As always, this kit includes all parts, all sockets, and complete instructions for ease of assembly. Because of our past experience with our 4K kit we suggest that you order early. All orders will be shipped on a strict first come basis. Dealers inquiries welcome on this item.

Kit shipped with 2 MHZ crystals for existing 500NS memory. Easily modified for faster RAM chips when the prices come down.

Z–80 Manual – $7.50 Separately.

Kit includes Zilog Manual and all parts.

JUMBO LED CAR CLOCK

$16.95 KIT

You requested it! Our first DC operated clock kit. Professionally engineered from scratch to be a DC operated clock. Not a makeshift kluge as sold by others. Features: Bowmar 4 digit .5 inch LED array, Mostek 50252 super clock chip, on board precision time base, 12 or 24 hour real time format, perfect for cars, boats, vans, etc. Kit contains PC Board and all other parts needed (except case). 50,000 satisfied clock kit customers cannot be wrong!

FOR ALARM OPTION ADD $1.50
FOR XFMR FOR AC OPERATION ADD $1.50

60 HZ CRYSTAL TIME BASE FOR DIGITAL CLOCKS
S.D. SALES EXCLUSIVE!

KIT FEATURES:

A. 60HZ output with accuracy comparable to a digital watch.
B. Directly interfaces with all MOS Clock Chips.
C. Super low power consumption. (1.5 ma typ.)
D. Uses latest MOS 17 stage divider IC.
E. Eliminates forever the problem of AC line glitches.
F. Perfect for cars, boats, campers, or even for portable clocks at ham field days.
G. Small Size, can be used in existing enclosures.

$5.95 or 2/$10.

KIT INCLUDES CRYSTAL, DIVIDER IC, PC BOARD PLUS ALL OTHER NECESSARY PARTS & SPECS

50HZ CRYSTAL TIME BASE KIT – $6.95

All the features of our 60HZ kit but has 50HZ output. For use with clock chips like the 50252 that require 50HZ to give 24 hour time format.

THIS MONTH'S SPECIALS!

SPECIAL *SPECIAL*

300.00 KHZ CRYSTAL – $1.50
8080A – CPU CHIP by AMD – $19.95
82S129 – 256 x 4 PROM – $2.50
N.S. 8865 OCTAL DARLINGTON DRIVERS
3 for $1.00
Z-80 – CPU by ZILOG – $69.95
MM5204 – 4K EPROM – $7.95

Prices in effect this month ONLY!

4K LOW POWER RAM BOARD KIT
THE WHOLE WORKS – $89.95

Imsai and Altair 8080 plug in compatible. Uses low power static 21L02–1 500ns. RAM's, which are included. Fully buffered, drastically reduced power consumption, on board regulated, all sockets and parts included. Premium quality plated thru PC Board.

SIGNETICS ANALOG MANUAL – $5.95

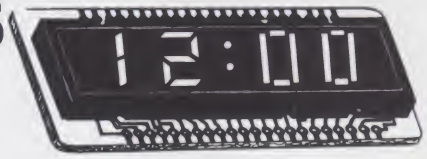
Just out! From the acknowledged leader in linear technology. Theory, applications, and specs. on op amps, timers, phase locked loops, etc. 637 pages.

A MUST For Any Technical Library!

STICK IT! in your clock in your DVM, etc.!

Huge Special Purchase Not Factory Seconds As sold by others!

$3.95



4 JUMBO .50" DIGITS ON ONE STICK! (with colons and AM/PM Indicator)

BUY 3 for $10.

BOWMAR 4 DIGIT LED READOUT ARRAY

The Bowmar Opto-Stick. The best readout bargain we have ever offered. Has four common cathode jumbo digits with all segments and cathodes brought out. Increased versatility since any of the digits may be used independently to fit your applications. Perfect for any clock chip, especially direct drive units like 50380 or 7010. Also use in freq. counters, DVM's, etc. For 12 or 24 hour format.

UP YOUR COMPUTER!
21L02–1 1K LOW POWER 500 NS STATIC RAM
TIME IS OF THE ESSENCE!

And so is power. Not only are our RAM'S faster than a speeding bullet but they are now very low power. We are pleased to offer prime new 21L02–1 low power and super fast RAM's. Allows you to STRETCH your power supply farther and at the same time keep the wait light off. **8 for $12.95**

HOUSE NO. TTL

7400 – 8/$1.00	7420 – 8/$1.00	74141 – 3/$1.00
7404 – 8/$1.00	7437 – 5/$1.00	74153 – 3/$1.00
7408 – 8/$1.00		7438 – 5/$1.00

Please specify that you are ordering House No. TTL

WESTERN DIGITAL UART

No. TR1602B. 40 pin DIP
This is a very powerful and popular part.
NEW–$6.95 with data
LIMITED QUANTITY

SLIDE SWITCH ASSORTMENT

Our best seller. Includes miniature and standard sizes, single and multi-position units. All new, first quality, name brand. Try one package and you'll reorder more. **SPECIAL 12/$1.**

RESISTOR ASSORTMENT

¼ W 5% and 10% PC leads. A good mix of values. 200/$2.

MOTOROLA POWER DARLINGTON

Back in Stock!

Like MJ3001. NPN 80V. 10A. HFE 6000 TYP. TO–3 case. We include a free 723C volt reg. with schematic for power supply. SPECIAL–$1.99

FAIRCHILD BIG LED READOUTS

A big .50 inch easy to read character. Now available in either common anode or common cathode. Take your pick. Super low current drain, only 5MA per segment typical.

FND 510 Common Anode
FND 503 Common Cathode
PRICE SLASHED! 59c each

CALL YOUR BANK AMERICARD OR MASTER CHARGE ORDER IN ON OUR CONTINENTAL UNITED STATES TOLL FREE WATTS:

1–800–527-3460
Texas Residents Call Collect
214/271-0022

TERMS:

Money Back Guarantee. No COD. Texas Residents add 5% tax. Add 5% of order for postage and handling. Orders under $10. add 75c. Foreign orders: US Funds ONLY!

S.D. SALES CO.
P.O. BOX 28810 K
Dallas, Texas 75228

BUILD YOUR LIBRARY



COMPONENT TESTERS

Build your own test equipment and save a bundle (and have a lot of fun). Volume 1 of the 73 Test Equipment Library shows you how to build and use transistor testers (8 of 'em), three diodes testers, 3 IC testers, 9 voltmeters and VTVMs, 8 ohmmeter, 3 inductance meters, 9 capacity meters, and a raft of other gadgets for checking temperature, crystals, Q, etc. $4.95

RADIO FREQUENCY TESTERS

This is of more interest to hams and CBers . . . test equipment you can build for checking out transmitters and receivers: signal generators, noise generators, crystal calibrators, GDOs, dummy loads . . . things like that. This is Volume 3 of the 73 Test Equipment Library (Prepublication offer) $4.95

AUDIO FREQUENCY TESTERS

If you're into audio . . . such as digital cassette recording, RTTY, Baudot vs ASCII, SSTV, SSB, Touchtone or even hi-fi . . . you'll want to have this book full of home built test equipment projects. Volume II (Prepublication offer) $4.95

MICROCOMPUTER DICTIONARY

Over 5000 definitions and explanations of terms and concepts (704 pages) relating to microprocessors, microcomputers and microcontrollers. There are also separate appendices on: programmable calculators; math and statistics definitions; flowchart symbols and techniques; binary number systems and switching theory; symbol charts and tables; summaries of BASIC FORTRAN and APL. In addition there is a comprehensive electronics/computer abbreviations and acronyms section. $15.95





HOBBY COMPUTERS ARE HERE

If you (or a friend) want to come up to speed on how computers work . . . hardware and software . . . this is an excellent book. It starts with the fundamentals and explains the circuits, the basics of programming, along with a couple TVT construction projects, ASCII-Baudot, etc. This book has the highest recommendations as a teaching aid for newcomers. $4.95

THE NEW COMPUTERS

This book takes over where the previous book leaves off. This, like the other, is a collection of reprints from recent issues of 73 Magazine (you've been missing a lot of very valuable data). This is one of the easiest ways to really understand how micros work and how to use them. Written entirely by hobbyists (prepublication offer). $4.95

What To Do After You Hit Return

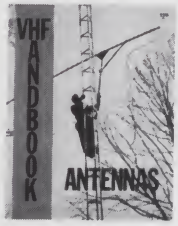
PCC's first book of computer games . . . 48 different computer games you can play in BASIC . . . programs, descriptions, muchly illustrated. Lunar landing, Hammurabi, King, Civil 2, Qubic 5, Taxman, Star Trek, Crash, Market, etc. $6.95 pp.



VHF ANTENNA HANDBOOK

The NEW VHF Antenna Handbook details the theory, design and construction of hundreds of different VHF and UHF antennas . . .

A practical book written for the average amateur who takes joy in building, not full of complex formulas for the design engineer. Packed with fabulous antenna projects you can build. $4.95



NOVICE STUDY GUIDE

This is the most complete Novice study guide available. It is brand new. This is not only invaluable for anyone wanting to get started in amateur radio, but also it is about the only really simple book on the fundamentals of electricity and electronics. And without your fundamentals down pat, how can you go on to really understand and work with computers? First things first. $4.95





BASIC . . . *by Bob Albrecht, etc.*

Self-teaching guide to the computer language you will need to know for use with your microcomputer. 324 pages. $4.95 pp.

Computer Programming Handbook

A complete guide to computer programming and data processing. Includes many worked out examples and history of computers. **$8.95**

WEATHER SATELLITE HANDBOOK

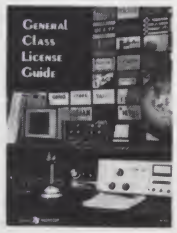
Simple equipment and methods for getting good pictures from the weather satellite. Antennas, receivers, monitors, facsimile you can build, tracking, automatic control (you don't even have to be home). Dr. Taggart WB8DQT $4.95.





GENERAL CLASS STUDY GUIDE

This book takes over on theory where the Novice book leaves off. You'll need to know the electronic theory in this to work with computers and you'll not find an easier place to get the information. It will also make getting your Tech or General license a breeze . . . then you can get on the ham repeaters and interconnect your micro with others. $5.95





RF AND DIGITAL TEST EQUIPMENT YOU CAN BUILD

RF burst, function, square wave generators, variable length pulse generators — 100 kHz marker, i-f and rf sweep generators, audio osc, af/rf signal injector, 146 MHz synthesizer, digital readouts for counters, several counters, prescaler, microwavemeter, etc. 252 pages. $5.95.



101 GAMES IN BASIC

Okay so once you get your computer up and running in BASIC, then what? Then you need some programs in BASIC, that's what. This book has 101 games for you, from very simple to real buggers. You get the games, a description of the games, the listing to put in your computer and a sample run to show you how they work. $7.50 pp.

SSTV HANDBOOK

This excellent book tells all about it, from its history and basics to the present state of the art techniques. Contains chapters on circuits, monitors, cameras, color SSTV, test equipment and much more.
Hardbound $7 Softbound $5



TTL COOKBOOK

by Donald Lancaster. Explains what TTL is, how it works, and how to use it. Discusses practical applications, such as a digital counter and display system, events counter, electronic stopwatch, digital voltmeter, and a digital tachometer. 336 pages; 5½ x 8½; softbound. **$8.95**



KILOBAUD MAGAZINE • PETERBOROUGH NH 03458 • USE ORDER CARD INSIDE BACK COVER!

7400N TTL

SN7400N	16	SN7459A	25		
SN7401N	16	SN7460N	22	SN74154N	1 00
SN7402N	21	SN7470N	45	SN74155N	99
SN7403N	16	SN7472N	39	SN74156N	99
SN7404N	18	SN7473N	37	SN74157N	99
SN7405N	24	SN7474N	32	SN74160N	1 25
SN7406N	20	SN7475N	50	SN74161N	99
SN7407N	29	SN7476N	32	SN74163N	99
SN7408N	25	SN7479N	5.00	SN74164N	1.10
SN7409N	25	SN7480N	50	SN74165N	1.10
SN7410N	18	SN7482N	98	SN74166N	1.25
SN7411N	30	SN7483N	70	SN74167N	5.50
SN7412N	33	SN7485N	89	SN74170N	2.10
SN7413N	45	SN7486N	39	SN74172N	8.95
SN7414N	70	SN7488N	3 50	SN74173N	1 50
SN7416N	35	SN7489N	2 49	SN74174N	1 25
SN7417N	35	SN7490N	45	SN74175N	99
SN7420N	21	SN7491N	75	SN74176N	90
SN7421N	33	SN7492N	49	SN74177N	90
SN7422N	49	SN7493N	49	SN74180N	99
SN7423N	37	SN7494N	79	SN74181N	2.49
SN7425N	29	SN7495N	79	SN74182N	95
SN7426N	29	SN7496N	89	SN74184N	1.95
SN7427N	37	SN7497N	4.00	SN74185N	2.20
SN7429N	42	SN74100N	1.00	SN74186N	15.00
SN7430N	26	SN74107N	39	SN74187N	6.00
SN7432N	31	SN74121N	39	SN74188N	3.95
SN7437N	27	SN74122N	39	SN74190N	1.19
SN7438N	27	SN74123N	50	SN74191N	1 25
SN7439N	25	SN74125N	60	SN74192N	89
SN7440N	15	SN74126N	60	SN74193N	89
SN7441N	89	SN74132N	1 09	SN74194N	1 25
SN7442N	59	SN74136N	95	SN74195N	75
SN7443N	75	SN74141N	1.15	SN74196N	1.25
SN7444N	75	SN74142N	4 00	SN74197N	75
SN7445N	75	SN74143N	4 50	SN74198N	1.75
SN7446N	81	SN74144N	4 50	SN74199N	1.75
SN7447N	69	SN74145N	1.15	SN74200N	5.59
SN7448N	79	SN74147N	2.35	SN74279N	.90
SN7450N	26	SN74148N	2.00	SN74251N	1 79
SN7451N	27	SN74150N	1.00	SN74284N	6.00
SN7453N	27	SN74151N	79	SN74285N	6 00
SN7454N	20	SN74153N	89	SN74367N	75

MANY OTHERS AVAILABLE ON REQUEST
20% Discount for 100 Combined 7400's

CMOS

CD4000	25	CD4035	1.85	74C04N	75
CD4001	25	CD4040	2 45	74C10N	65
CD4002	25	CD4042	1 90	74C20N	65
CD4006	2 50	CD4044	1 50	74C30N	65
CD4007	25	CD4046	2 51	74C42N	2 15
CD4009	59	CD4047	2 75	74C73N	1 50
CD4010	59	CD4049	79	74C74	1 15
CD4011	25	CD4050	79	74C90N	3 00
CD4012	25	CD4049	79	74C95N	2 00
CD4013	47	CD4050	79	74C107N	1 25
CD4016	56	CD4051	2 95	74C151	2 90
CD4017	1 35	CD4053	2 95	74C154	4 00
CD4019	55	CD4060	3 25	74C157	2 15
CD4020	1 49	CD4066	1 75	74C160	3 25
CD4022	25	CD4069	45	74C161	3 25
CD4023	25	CD4071	45	74C163	3 00
CD4024	1 50	CD4081	45	74C164	3 25
CD4025	25	CD4511	2 50	74C173	2 60
CD4027	69	CD4518	2 50	74C193	2 75
CD4028	1 65	CD4566	3.00	74C195	2 75
CD4029	2 90	74C00N	39	MC404L	4 50
CD4030	65	74C02N	55	MC1401E	56

LINEAR

LM300H	80			LM1310N	2.95
LM301H	35	LM370N	1 1 5	LM1351N	1 65
LM301CN	35	LM373N	3 25	LM1414N	1.75
LM302H	75	LM377N	4 00	LM1458C	65
LM304H	1 00	LM380N	1.39	LM1496N	95
LM305H	95	LM380N	1 05	LM2111N	1 95
LM307CN	35	LM377N	4 00	LM2901N	2 95
LM308H	1 00	LM380N	1.39	LM3065N	89
LM308CN	1.00	LM380N	1 05	LM3900N	55
LM309H	1 10	LM381N	1.79	LM3905N	60
LM309K	99	LM382N	1 79	LM3909	1 25
LM310CN	1 15	NE501K	8.00	LM5556N	1 85
LM311H	90	NE510A	6 00	MC5558V	1 00
LM311N	90	NE531H	3 00	LM5558N	90
LM318CN	1 50	NE536T	6 00	LM7525N	90
LM319N	1 30	NE540L	6 00	LM7535N	1 25
LM320K-5	1.35	NE550N	79	8038B	4 95
LM320K-5 2	1.35	NE555V	59	LM7540	49
LM320K-12	1 35	NE560B	5 00	75451CN	39
LM320K-15	1.35	NE561B	5 00	75452CN	39
LM320T-5	1 75	NE562B	5 00	75453CN	39
LM320T-5 2	1 75	NE565H	1 25	75454CN	39
LM320T-8	1 75	NE565N	1 75	75491CN	89
LM320T-12	1 75	NE566CN	1.25	75492CN	89
LM320T-15	1 75	NE567H	1 95	75494CN	89
LM320T-18	1 75	NE567V	1 50	**RCA LINEAR**	
LM320T-24	1 75	LM703CN	45	CA3013	2 15
LM323K-5	9 95	LM709H	29	CA3032	2 56
LM324N	1 80	LM709N	29	CA3035	2 48
LM339N	1 70	LM710N	79	CA3039	1.35
LM340K-5	1 95	LM711N	39	CA3046	1 30
LM340K-6	1 95	LM723H	55	CA3059	3 25
LM340K-8	1 95	LM723H	55	CA3060	3.25
LM340K-12	1 95	LM733N	1 00	CA3080	85
LM340K-15	1 95	LM739N	1.00	CA3081	2 00
LM340K-18	1 95	LM741CH	35	CA3082	2 00
LM340K-24	1 95	LM741CN	35	CA3083	1 60
LM340T-5	1.75	LM74114N	39	CA3086	85
LM340T-6	1.75	LM747H	79	CA3089	3.75
LM340T-8	1.75	LM747N	79	CA3091	10 20
LM340T-12	1.75	LM748H	39	CA3123	2 15
LM340T-15	1.75	LM748N	39	CA3130	1.39
LM340T-18	1.75	LM1303N	90	CA3140	1.25
LM340T-24	1.75	LM1304N	1 19	CA3600	1.75
LM350N	1 00	LM1305N	1 40	RC4194	5 95
LM351CN	65	LM1307N	85	RC4195	3 25

74LS00 TTL

74LS00	39			74LS139	1.95
74LS02	39	74LS74	65	74LS151	1 55
74LS03	39	74LS75	79	74LS153	1 89
74LS04	45	74LS76	65	74LS157	1 55
74LS05	45	74LS83	2 19	74LS162	2 25
74LS08	39	74LS85	2.49	74LS163	2 25
74LS10	39	74LS86	65	74LS164	2 25
74LS13	79	74LS90	1 25	74LS175	1.95
74LS14	2 19	74LS92	1 25	74LS181	3 69
74LS20	39	74LS93	1 25	74LS190	2 85
74LS26	49	74LS95	2 19	74LS191	2 85
74LS27	45	74LS96	1 89	74LS192	2 85
74LS28	49	74LS107	65	74LS193	2 85
74LS30	39	74LS109	.65	74LS194	2 25
74LS32	45	74LS112	65	74LS195	2 25
74LS40	49	74LS132	1 55	74LS257	1 89
74LS51	39	74LS136	65	74LS260	55
74LS55	39	74LS138	1 89	74LS279	79
74LS73	65			74LS670	5 95

CLOCK CHIPS

MM5309	6 Digit, BCD Outputs. Reset PIN	$9 95
MM5311	6 Digit, BCD Outputs. 12 or 24 Hour	4 95
MM5312	4 Digit. BCD Outputs. 1 PPS Output	4 95
MM5314	6 Digit. 12 or 24 Hour. 50 or 60 Hz	4 95
MM6316	4 Digit. Alarm. 1 PPS Output	6 95
MM5318	Video Clock Chip. For Use With MM5841	9 95
CT7001	6 Digit. Calendar. Alarm. 12 or 24 Hour	5 95

DATA HANDBOOKS

7400	Pin out & Description of 5400/7400 ICS	$2.95
CMOS	Pin out & Description of 4000 Series ICS	$2.95
Linear	Pin out & Functional Description	$2.95

ALL THREE HANDBOOKS $6.95

FAIRCHILD TECHNOLOGY KITS FAIRCHILD



- Complete Specifications on back of each kit
- Packaged for WALL DISPLAY APPEARANCE
- Dealer's Inquires Invited — Price List Available



DIGITS

FTK0001	0.5" High Common Cathode Digit	$1 00
FTK0002	0.5" High Common Anode Digit	1 00
FTK0003	.357" High Common Cathode Digit	.75
FTK0004	0.8" High Common Cathode Digit	2 00
FTK0005	0.8" High Common Anode Digit	2 00

0.8" HIGH DISPLAY ARRAYS

FTK0010	12 Hour. 3½ Digit Clock Display	7 00
FTK0011	24 Hour. 4 Digit Clock Display	8 00

LED LAMPS

FTK0020	10 Red LED Lamps	1 00
FTK0021	5 Mixed Colored LED Lamps	1 00
FTK0022	10 LED Mounting Clips	1 00
FTK0023	5 Three Piece LED Mounting Adapters	1 00

PHOTO TRANSISTORS

FTK0030	5 Flat Lens Photo Transistors	1 00
FTK0031	5 Round Lens Photo Transistors	1 00
FTK0032	3 Flat Lens Photo Darlingtons	1 00
FTK0033	3 Round Lens Photo Darlingtons	1 00

PHOTO ARRAYS

FTK0040	9-Element Tape Reader Array	16 00
FTK0041	12-Element Card Reader Array	24 00
FTK0042	Reflective Opto Coupler	4 00

COUPLERS

FTK0050	3 General Purpose Opto Couplers	1 00
FTK0051	Darlington Opto Coupler	1 00

MOS CLOCK CIRCUITS

FTK0400	Digital Clock/Calendar Circuit (FCM7001)	7 00
FTK0401	Digital Clock/Calendar with BCD Outputs (FCM7002)	7 00
FTK0402	Direct Drive Digital Clock Circuit with AC Output (FCM3817A)	5 00
FTK0403	Direct Drive Digital Clock Circuit with DC Output (FCM3817D)	5 00
FTK0405	Direct Drive Digital Clock/Calendar Circuit (FCM7015)	6 00

KITS

FTK0106	Automobile Clock Kit	40 00

DISCRETE LEDS

	.125" dia.					.200" dia.					.90" dia.		
XC209	Red	10/$1				XC111	Red	10/$1			XC111	Red	10/$1
XC209	Green	4/$1				XC111	Green	4/$1			XC111	Green	4/$1
XC209	Orange	4/$1				XC111	Yellow	4/$1			XC111	Orange	4/$1
	.200" dia.			.185" dia.			.200" dia.			.085" dia.			
XC22	Red	10/$1	XC526	Red	10/$1	XC556	Red	10/$1	MV50				
XC22	Green	4/$1	XC526	Green	4/$1	XC556	Green	7/$1	.085 dia Micro red LED				
XC22	Yellow	4/$1	XC526	Yellow	4/$1	XC556	Yellow	7/$1	6/$1				
XC22	Orange	4/$1	XC526	Orange	4/$1	XC556	Orange	7/$1					
SSL-22	RT	4/$1	XC526	Clear	4/$1	XC556	Clear	7/$1					

DISPLAY LEDS

DL707 DL33B

TYPE	POLARITY	HT		TYPE	POLARITY	HT	
MAN 1	Common Anode	270	2 95	MAN 3620	Common Anode-orange	300	1 75
MAN 2	5 x 7 Dot Matrix	300	4 95	MAN 3640	Common Cathode-orange	300	1 75
MAN 3	Common Cathode	125	.39	MAN 4710	Common Anode-Red	400	1 95
MAN 4	Common Cathode	187	1 95	DL701	Common Anode-red	300	.95
MAN 7	Common Anode	300	1 25	DL704	Common Cathode	300	.95
MAN 7G	Common Anode-green	300	1 95	DL707	Common Anode	300	.95
MAN 7Y	Common Anode-yellow	300	1 95	DL 728	Common Cathode	500	1.00
MAN 72	Common Anode-green	300	1.75	DL 747	Common Anode	600	2 25
MAN 64	Common Anode-red	400	1 75	DL 750	Common Cathode	600	2 49
MAN 72	Common Anode	300	1 25	DL 33B	Common Cathode	110	.50
MAN 74	Common Cathode	300	1 50	FND70	Common Cathode	250	.75
MAN 82	Common Anode-yellow	300	1 75	FND503	Common Cathode	500	1 00
MAN 84	Common Cathode-yellow	300	1 75	FND507	Common Anode	500	1 00

IC SOLDERTAIL — LOW PROFILE (TIN) SOCKETS

	1-24	25-49	50-100		1-24	25-49	50-100
8 pin	$ 17	16	15	24 pin	$ 38	37	36
14 pin	20	19	18	28 pin	45	44	43
16 pin	22	21	20	36 pin	60	59	58
18 pin	29	28	27	40 pin	63	62	61
22 pin	37	36	35				

SOLDERTAIL STANDARD (TIN)

14 pin	$ 27	25	24	28 pin	$ 99	90	81
16 pin	30	27	25	36 pin	1 39	1 26	1 15
18 pin	35	32	30	40 pin	1 59	1 45	1 30
24 pin	49	45	42				

SOLDERTAIL STANDARD (GOLD)

8 pin	$ 30	27	24	24 pin	$ 70	63	57
14 pin	35	32	29	28 pin	1 10	1 00	90
16 pin	38	35	32	36 pin	1 75	1 40	1 26
18 pin	52	47	43	40 pin	1 75	1 59	1 45

WIRE WRAP SOCKETS (GOLD) LEVEL #3

10 pin	$ 45	41	37	24 pin	$1 05	95	85
14 pin	39	38	37	28 pin	1 40	1 25	1 10
16 pin	43	42	41	36 pin	1 59	1 45	1 30
18 pin	75	68	62	40 pin	1 75	1 55	1 40

WIRE WRAP TOOL Part Number WSU 30 $5.95 ea.

(FOR 30 AWG WIRE)



WIRE WRAP WIRE

30 AWG — 25 ft. min. - $2.10 50 ft. - $2.75 100 ft. - $3.50 1000 ft. - $24.00

SPECIFY COLOR — White - Yellow - Red - Green - Blue - Black

50 PCS. RESISTOR ASSORTMENTS $1.75 PER ASST.

		10 OHM	12 OHM	15 OHM	18 OHM	22 OHM	
ASST. 1	5 ea.	27 OHM	33 OHM	39 OHM	47 OHM	56 OHM	1/4 WATT 5% - 50 PCS.
		68 OHM	82 OHM	100 OHM	120 OHM	150 OHM	
ASST. 2	5 ea.	180 OHM	220 OHM	270 OHM	330 OHM	390 OHM	1/4 WATT 5% - 50 PCS.
		470 OHM	560 OHM	680 OHM	820 OHM	1K	
ASST. 3	5 ea.	1.2K	1.5K	1.8K	2.2K	2.7K	1/4 WATT 5% - 50 PCS.
		3.3K	3.9K	4.7K	5.6K	6.8K	
ASST. 4	5 ea.	8.2K	10K	12K	15K	18K	1/4 WATT 5% - 50 PCS.
		22K	27K	33K	39K	47K	
ASST. 5	5 ea.	56K	68K	82K	100K	120K	1/4 WATT 5% - 50 PCS.
		150K	180K	220K	270K	330K	
ASST. 6	5 ea.	390K	470K	560K	680K	820K	1/4 WATT 5% - 50 PCS.
		1M	1.2M	1.5M	1.8M	2.2M	
ASST. 7	5 ea.	2.7M	3.3M	3.9M	4.7M	5.6M	1/4 WATT 5% - 50 PCS.

ASST. 8R Includes Resistor Assortments 1-7 (350 PCS.) $10.95 ea.

$5.00 Minimum Order — U.S. Funds Only
California Residents — Add 6% Sales Tax

Spec Sheets - 25¢ — Send 24¢ Stamp for 1977 Catalog
Dealer Discount Available — Request Pricing

NOW OPEN SATURDAYS

James ELECTRONICS

1021-A HOWARD AVE., SAN CARLOS, CA. 94070
PHONE ORDERS WELCOME — (415) 592-8097
All Advertised Prices Good Thru January

MINIATURE TOGGLE

DPDT				JMT	1	2 to 9	10-29	30-99
	ON	OFF	ON	221	2 95	2 55	1 87	1 70
	ON	NONE	ON	223	2 95	2 15	1 58	1 43
	ON	OFF	ON	121	2 35	1 95	1 43	1 30
	ON	NONE	ON	123	2 05	1 65	1 21	1 10



PUSH BUTTON

SPDT SERIES PB PB1

Maintained Action Switch
Push To Make Contact And Push Again To Reserve Action

Momentary Action Switch
Normally Open or Normally Closed If pin N C circuit use terminals 2 & 1 for N O 2 & 3 for N C and N O 1 2 & 3

PB-123 $1.75

PB-126 $1.75

5 AMP RATINGS



THUMBWHEEL SWITCHES

THUMBWHEEL SWITCH ONLY

Part No	Description	Price
SF 12	Single Pole 10 Position	$2 50
SR 12	Decimal	3 00
SF 21	10 Position BCD only	2 50
SR 21		3 00

Ordering: Order desired switch or switches and add necessary accessories for your particular application

SF Front Mount SR Rear Mount

SERIES SF Front Mount Assembly



SERIES SR Rear Mount Assembly



ACCESSORIES

Part No	Description	Price
SF EP	End Plate (Pair)	50
SF DP	Divider Plate (each)	40
SF BB	Blank Body (each)	40
SF HB	Half Body (each)	40

ACCESSORIES

Part No	Description	Price
SR EP	End Plates (pair)	$ 50
SR DP	Divider Plate (each)	40
SR BB	Blank Body (each)	40
SR HB	Half Body (each)	40

DIP SWITCH



These switches feature seven SPST slide switches in a molded dip. They are ideally suited for microprocessor applications

$1.95

CLIPLITE !! 4/$1.00



NEW LED MOUNTING SYSTEM

- CLIPLITE mounts from the front of the Panel in a .250 hole on 3/8" centers. Panel thickness from 1/16" to 1/8".
- CLIPLITE equalizes and increases the brightness of commonly used wide beam LEDS.
- CLIPLITE is to be used with XC556 LEDS.
- Specify colors—red, green, amber, or yellow when ordering.

ZENERS — DIODES — RECTIFIERS

TYPE	VOLTS	W	PRICE	TYPE	VOLTS	W	PRICE
1N746	3 3	400mm	4 1 00	1N4005	600 PIV	1 AMP	10 1 00
1N751A	5 1	400m	4 1 00	1N4006	800 PIV	1 AMP	10 1 00
1N752	5 6	400m	4 1 00	1N4007	1000 PIV	1 AMP	10 1 00
1N753	6 2	400m	4 1 00	1N3600	50	200m	6 1 00
1N754	6 8	400m	4 1 00	1N4148	.75	10m	15 1 00
1N959	8 2	400m	8 1 00	1N4154	.35	10m	12 1 00
1N965B	15	400m	4 1 00	1N4305	.75	25m	20 1 00
1N5232	5 6	500m	28	1N4734	5 6	1w	28
1N5234	6 2	500m	28	1N4735	6 2	1w	28
1N5235	6 8	500m	28	1N4736	6 8	1w	28
1N5236	7 5	500m	28	1N4738	8 2	1w	28
1N456	25	40m	6 1 00	1N4742	12	1w	28
1N458	150	7m	6 1 00	1N4744	15	1w	28
1N485A	180	10m	6 1 00	1N1183	50 PIV	35 AMP	1 60
1N4001	50 PIV	1 AMP	12 1 00	1N1184	100 PIV	35 AMP	1 70
1N4002	100 PIV	1 AMP	12 1 00	1N1185	150 PIV	35 AMP	1 50
1N4003	200 PIV	1 AMP	12 1 00	1N1186	200 PIV	35 AMP	1 80
1N4004	400 PIV	1 AMP	12 1 00	1N1188	400 PIV	35 AMP	3 00

SCR AND FW BRIDGE RECTIFIERS

C36D	15A @ 400V	SCR	$1 95
C38M	35A @ 200V	SCR	1 95
2N2328	1 6A @ 200V	SCR	50
MDA 980-1	25A @ 50V	FW BRIDGE REC	1 95
MDA 980-3	25A@ 200V	FW BRIDGE REC	1 95

TRANSISTORS

MPS A05	5/$1 00			PN3567	3/$1 00	PN4249	4/$1 00
MPS A06	5/$1 00			PN3568	4/$1 00	PN4250	4/$1 00
2N2219A	3/$1 00			PN3569	4/$1 00	2N4400	4/$1 00
2N2221	4/$1 00			2N3704	5/$1 00	2N4401	4/$1 00
2N2222A	5/$1 00			2N3705	5/$1 00	2N4402	4/$1 00
2N2369	5/$1 00			2N3706	5/$1 00	2N4403	4/$1 00
2N2369A	4/$1 00			2N3707	5/$1 00	2N4409	5/$1 00
FN2415	5/$1 00			2N3711	5/$1 00	2N5086	4/$1 00
2N2484	4/$1 00			2N3724	$ 65	2N5087	4/$1 00
2N2906A	4/$1 00			2N3725	$1 00	2N5088	4/$1 00
2N2907A	5/$1 00			2N3903	5/$1 00	2N5089	4/$1 00
2N2925	5/$1 00			2N3904	4/$1 00	2N5129	5/$1 00
2N3053	2/$1 00			2N3905	4/$1 00	2N5138	5/$1 00
2N3055	$ 89			2N3906	4/$1 00	2N5139	5/$1 00
MJE3055	$1 00			2N4013	3/$1 00	2N5209	5/$1 00
MJE2955	$1 25			2N4014	3/$1 00	2N5951	5/$1 00
2N3392	5/$1 00			2N4123	10/$1 00	C106B1SCR	2/$1 00
2N3398	5/$1 00					2N5432	32 00

CAPACITOR CORNER

50 VOLT CERAMIC DISC CAPACITORS

	1-9	10-49	50-100		1-9	10-49	50-100
10 pf	05	.04	03	001µF	05	04	035
22 pf	05	04	03	0047µF	05	04	035
47 pf	05	04	03	01µF	05	04	035
100 pf	05	04	03	022µF	06	05	04
220 pf	05	04	03	047µF	06	05	04
470 pf	05	04	035	1µF	12	09	075

100 VOLT MYLAR FILM CAPACITORS

001mf	12	10	07	022mf	13	11	08
0022	12	10	07	047mf	21	17	13
0047mf	12	10	07	1mf	27	23	17
01mf	12	10	07	22mf	33	27	22

+20% DIPPED TANTALUMS (SOLID) CAPACITORS

1/35V	28	23	17	1 5/35V	30	26	21
15/35V	28	23	17	2 2/25V	31	27	22
22/35V	28	23	17	3 3/25V	31	27	22
33/35V	28	23	17	4 7/25V	32	28	23
47/35V	28	23	17	6 8/25V	36	31	25
68/35V	28	23	17	10/25V	40	35	29
1 0/35V	28	23	17	15/25V	63	50	40

MINIATURE ALUMINUM ELECTROLYTIC CAPACITORS

	Axial Lead				Radial Lead		
47/50V	15	13	10	47/25V	15	13	10
1 0/50V	16	14	11	47/50V	16	14	11
3 3/50V	15	13	10	1 0/16V	15	13	10
4 7/25V	16	14	12	1 0/25V	16	14	11
10/25V	15	13	10	1 0/50V	16	14	11
10/50V	16	14	12	4 7/16V	15	13	10
22/25V	17	15	12	4 7/25V	15	13	10
22/50V	24	20	18	4 7/50V	16	14	11
47/25V	19	17	15	10/16V	14	12	09
47/50V	25	21	19	10/25V	15	13	10
100/25V	24	20	18	10/50V	16	14	12
100/50V	35	30	28	47/50V	24	21	19
220/25V	32	28	25	100/16V	19	15	14
220/50V	45	41	38	100/25V	24	20	18
470/25V	33	29	27	100/50V	35	30	28
1000/16V	55	50	45	220/16V	23	17	16
2200/16V	70	62	55	470/25V	31	28	26

Subscribe To 73? ME?



This may come as a big surprise to you, but there are a lot of non-ham readers of 73. The barrage of computer articles in the magazine during the last year (over 300 pages of articles on hobby computers . . . how many other magazines can claim that much?) has not entirely escaped the eyes of all computerists.

Yes, 73 certainly does cover the 25 or so hobbies which are classed together as amateur radio . . . radioteletype (RTTY), slow scan television (SSTV), DXing (contacting foreign countries), FAX (facsimile transmission), FM and repeaters, satellite use, moonbouncing, contests, experimenting, home construction, and even rag chewing. There are articles on building and using test equipment, on timers, weather satellite systems, wind speed measuring, and hundreds of other related and unrelated subjects.



73 is the largest of the ham magazines . . . the January 1977 issue ran 55 articles and was over 200 pages. Add to all of that the irreverence of Wayne Green and his editorials and you have a magazine that most readers read from cover to cover every month . . . no matter how many other magazines they get.

Do you really want to continue to miss all those computer articles?

cut here or copy

SUBSCRIPTION TO 73 MAGAZINE ☐ $15 One year* ☐ $25 Three years* ☐ $149 Life

Name _____ Call (if any) _____

Address _____

City _____ State _____ Zip _____

*U.S. & Canada only. All others please add $2/year foreign postage.

☐ Cash enclosed ☐ Check enclosed ☐ Money order enclosed

Charge to:

☐ Master Charge ☐ BankAmericard ☐ American Express

Credit Card # _____ Interbank # _____

Expiration date _____ Signature _____

☐ Bill me direct

YOU MAY USE THE CARD INSIDE THE BACK COVER FOR THIS ORDER.

73 MAGAZINE, PETERBOROUGH NH 03458 USA 1-77

Toll Free subscription numbers: 800-258-5473 or 800-251-6771

kilobaud

index of advertisers

A-2	Aldelco 138
A-12	Ancrona 131
A-27	A-OK Electronics 94
A-18	AST/Servo Systems 74, 107
C-5	Communications Electronics 91
C-28	The Computer Corner 121
C-29	Computer Mart Stores 121
C-30	Computer Mart of N.J. 94
C-27	Computer Warehouse Store 113
C-31	Computer Workshop 121
C-20	C & S Marketing Assoc. 49
D-12	Digital Group 75
E-13	ECD Corp. C III
E-14	Electronic Eng. & Production 138
E-15	Erickson, B 138
G-4	Godbout 71
H-10	Hufco 135
J-1	James Electronics 143
L-3	Logical Services 121
M-2	Meshna 139
M-12	Micro Software Specialists 98
M-4	MITS C IV, 25-33
N-1	National Multiplex 1
O-5	OK Machine & Tool 13 & 15
O-1	Ohio Scientific 7
O-3	Optoelectronics 85
P-8	Personal Computing Co. 95
P-11	Proko Electronics Shoppe 123
Q-4	Quality Security Systems 121
R-12	Rainbow Computing 37
R-7	Rondure Co. 133
S-1	Scelbi 94
S-26	Scientific Research 48
S-2	S. D. Sales Co. 140 & 141
S-22	Seals Electronics 61
S-4	Slep Electronics 137
S-6	Southwest Technical Products C II
S-11	A. F. Stahler Co. 37
S-12	Suntronix Co. 111
T-11	Tarbell Electronics 103
T-12	Technical Systems Consultants 113
T-13	Teletypewriter Comm. Spec. 23
T-1	Tri-Tek 117
V-8	Vector Electronic Co. 98
V-9	Veras Systems 43
W-13	Wasatch Semiconductor Products 98
W-5	Wave Mate 2
W-14	WWW Enterprises 113

FROM KILOBAUD . . .

Kilobaud Sweeps 38 & 39
Kilobaud Life Subs 103
73 Tapes 129
Kilobaud Library 142
73 Sub 144

PETERBOROUGH NH 03458

kilobaud reader service

Circle appropriate Reader Service # for desired company brochures, data sheets or catalogs and mail to Kilobaud Magazine, Peterborough NH 03458. Include your zip code, please. Send money directly to advertisers. LIMIT: 25 requests.

A-2	C-27	H-10	O-1	S-26	T-12	KILOBAUD
A-12	C-31	J-1	O-3	S-2	T-13	
A-27	C-20	L-3	P-8	S-22	T-1	
A-18	D-12	M-2	P-11	S-4	V-8	
C-5	E-13	M-12	Q-4	S-6	V-9	
C-28	E-14	M-4	R-12	S-11	W-13	
C-29	E-15	N-1	R-7	S-12	W-5	
C-30	G-4	O-5	S-1	T-11	W-14	

☐ Subscriber ☐ Newsstand buyer

PLEASE PRINT OR TYPE

Name _____

Address _____

City _____ State _____ Zip _____

Coupon Expires in 60 Days 1/77

PLACE STAMP HERE

kilobaud

PETERBOROUGH NH 03458

Att: Mail Order

I want to subscribe!

Send me Kilobaud for:

☐ 1 year — ~~$15.00*~~ $12.00* SPECIAL!
☐ 3 years — ~~$35.00*~~ $25.00* SPECIAL!
☐ Life Subscription — $150.00

☐ Renewal
☐ New sub.

Enclosed $_____ ☐ Cash ☐ Check ☐ Money Order
☐ American Express ☐ BankAmericard ☐ Master Charge

Credit card # _____ Interbank # _____

Expiration date _____ Signature _____

☐ Bill me directly Signature _____

Name _____

Address _____

City _____ State _____ Zip _____

*U.S. & Canada only! All others please add $2/year foreign postage. 1/77

FIRST CLASS
Permit No. 1024
Peterborough NH 03458

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN UNITED STATES

POSTAGE WILL BE PAID BY

kilobaud
PETERBOROUGH NH 03458

Att: Reader Service

PLEASE PRINT OR TYPE
Please send me the following Kilobaud products:

Quantity	Description	Unit Price	Total

Enclosed $_____ ☐ Cash ☐ Check ☐ Money Order
Total for Order
Bill: ☐ American Express ☐ BankAmericard ☐ Master Charge
Credit Card # _____ Interbank # _____
Expiration date _____ Signature _____
Name _____
Address _____
City _____ State _____ Zip _____ 1/77

PLACE STAMP HERE

kilobaud
PETERBOROUGH NH 03458

Att: Subscriptions

books,etc.

Test Equipment Library
- Vol I – Component Testers $4.95
- Vol II – Audio Frequency Testers $4.95
- Vol III – Radio Frequency Testers $4.95

VHF Antenna Handbook $4.95
Weather Satellite Handbook $4.95
RF & Digital Test Equipment $5.95
SSTV Handbook Soft $5, Hard $7
TTL Cookbook $8.95
Novice Study Guide $4.95
General Class Study Guide $5.95

BASIC $4.95
Microcomputer Dictionary $15.95
Computer Programming Handbook $8.95
101 Games in Basic $7.50
What To Do After
- You Hit Return $6.95

Hobby Computers Are Here $4.95
The New Computers $4.95
Novice Theory Tapes
- set of four – $15.95

Code Tapes
- 5, 6, 14, 21 WPM – $4.95 each

kilobaud

moving?

Mail us your new address as soon as possible to insure uninterrupted delivery of your magazines.

You may use the pre-printed order card for your change of address:

1. Write in "description" column "MY OLD ADDRESS WAS:"
2. Attach old label if available . . . otherwise print *clearly* the old address (don't forget the zip code).
3. Clearly print your name & new address in the spaces provided at the bottom of the card.



Key Into Maxi-Power @ Micro-Price

Micromind is an incredibly flexible, omplete and expandable, hardware/ oftware, general purpose computer ystem. You won't outgrow it.

Hardware includes an 80 key, software-efinable keyboard, I/O interface board, 500A-series microprocessor (powerful nough for advanced computing), a high-etail graphics and character display rocessor, power supply, rf modulator, nd connections for up to 4 tape recorders us TV or monitor. An interconnect bus





ermits 15 additional microprocessors, arallel processing and vastly increased mputing power.

System software—including ECD's own otsoBASIC high level language, on vanced error-correcting tape cassettes provides a word processing editor, a powerful assembler, a debugger, a file system, graphic routines, and peripheral handlers. We also include dynamic graphic games: Animated Spacewar and Life.

ECD's standard Micromind μM-65 supplies 8K bytes of memory. Additional 32K byte expansion boards and a mapping option give Micromind expandable access to 64 Megabytes. Utilizing software-controlled I/O channels, Micromind's advanced encoding techniques load data from ordinary tape recorders at 3200 bits per second.

Micromind comes to you ready-to-use, factory assembled and fully tested. Among microcomputers, it has the largest memory capacity and the fastest storage. You're looking at the work of the finest display processor on the market. You won't find a microcomputer with a more powerful CPU. You won't find a computer with a more flexible keyboard. You won't find anything to touch it at **$987.54.**





So, quit the kluge scene and key into Micromind. You'll be a main frame performer, with all the comforts of home. We're not fooling...this is the cat's μ!

ECD CORP.
196 Broadway, Cambridge, Mass. 02139
(617) 661-4400

ECD

Name _____

Address _____

City/State _____ Zip _____

☐ Fantastic! Check enclosed: **$987.54.** Shipping paid by ECD
☐ BankAmericard ☐ Master Charge Mass. Resident add 5% Sales Tax

_____ Expiration Date _____

Signature _____

☐ Send me your brochure.

7

Actual unretouched photographs.

The small wonder



of the micro-world

Measuring just 11" wide x 11" deep x 5" high, and weighing a mere 7 pounds, the Altair™ 680b is a complete, general-purpose computer.

The secret to this revolutionary, small computer is its CPU board. This double-sided board fits along the bottom of the Altair case and plugs directly into the front panel board. It contains the new 6800 microprocessor, 1,024 bytes of RAM memory, a 256 byte PROM monitor, provisions for 768 bytes of additional PROM or ROM, and a single Interface port with a Motorola AGIA serial interface adapter which can be configured either RS-232 or TTY. A five level Baudot interface option is also available.

The Altair 680b can be programmed from front panel switches, or it can be interfaced to a video display terminal, or teletype-writer. Three additional circuit boards can be plugged inside the Altair 680b for further memory and interface expansion. The first of these boards is a 16K static RAM memory board.

Software already developed includes Altair 680 BASIC with all the features of the 8K BASIC previously developed for the Altair 8800. These include Boolean operators, the ability to read or write a byte from any I/O port or memory location, multiple statements per line, and the ability to interrupt program execution and then continue after the examination of variable values. This software takes only 6.8K bytes of memory space and a copy is **included free with the purchase of the Altair 680 16K memory board.**

Other software includes a resident two-pass assembler and text editor. The Altair 680b is also compatible with Motorola 6800 software.

The Altair 680b is ideal for hobbyists who want a powerful computer system at an economic price. Altair 680b owners qualify for membership in the Altair Users Group, and like other Altair owners, they receive a complimentary subscription to **Computer Notes** and complete factory support.

NOTE: Altair is a trademark of MITS, Inc.

PRICES:

Altair 680b kit with complete, easy-to-understand assembly manual, operator's manual, and programming manual	$466
Assembled Altair 680b	$625
Altair 680b Turnkey model kit	$395
Expander Card 680MB (required to expand 680)	$ 24
Altair 680BSM 16K static RAM board kit with 680 BASIC	$685
Altair 680 BASIC when purchased separately	$200
Baudot option	$ 42

MAIL THIS COUPON TODAY

☐ Enclosed is a check for $_____

☐ BankAmericard # _____ or Master Charge # _____

☐ Altair 680b ☐ Kit ☐ Assembled ☐ Other (specify)

enclosed $8 for postage and handling (Continental U.S. Others subject to quote)

☐ Please send free information package

NAME _____

ADDRESS _____

CITY _____ STATE & ZIP _____

mits

2450 Alamo SE/Albuquerque, NM 87106, 505-243-7821

Price, specifications subject to change. Please allow up to 90 days for delivery